Vision-Aided Autonomous Precision Weapon Terminal Guidance Using a Tightly-Coupled INS and Predictive Rendering Techniques

THESIS

Jonathan W. Beich, Major, USAF

AFIT/GE/ENG/11-42

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GE/ENG/11-42

Vision-Aided Autonomous Precision Weapon Terminal Guidance Using a Tightly-Coupled INS and Predictive Rendering Techniques

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Electrical Engineering

Jonathan W. Beich, BSEE, MS Space Studies, MS Flight Test Engineering

Major, USAF

March 2011

AFIT/GE/ENG/11-42

Vision-Aided Autonomous Precision Weapon Terminal Guidance Using a Tightly-Coupled INS and Predictive Rendering Techniques

Jonathan W. Beich, BSEE, MS Space Studies, MS Flight Test Engineering
Major, USAF

Approved:

/signed/	7 March 2011
Lt Col M. Veth, PhD (Chairman)	date
/signed/	7 March 2011
Dr. J. Raquet (Member)	date
/signed/	7 March 2011
Dr. M. Pachter (Member)	date
/signed/	7 March 2011
Dr. R. Martin (Member)	date

Abstract

This thesis documents the development of the Vision-Aided Navigation using Statistical Predictive Rendering (VANSPR) algorithm which seeks to enhance the endgame navigation solution possible by inertial measurements alone. The impetus of the work is the design of a precision weapon that does not rely on the Global Positioning System, functions autonomously, thrives in complex three-dimensional environments, is impervious to jamming, and can perform adequately with incomplete information.

Before the algorithm can be used, virtual world models are constructed of the target environment and constituent objects. These models are designed to be representative in size, shape, and texture by utilizing physical measurement and digital photographs of object surfaces. Eight data collection flights were executed at the Air Force Test Pilot School by modifying a C-12C aircraft with a high-quality scientific-grade digital camera, an inertial navigation system, and supporting hardware and software and flying it against five target environments at various aspects. The test algorithm employs a nonlinear Unscented Kalman Filter (UKF) which seeks to determine navigation errors by comparing real images captured by the test camera with a collection of statistically significant virtual images.

Results indicate that the VANSPR algorithm has the potential to be a viable method of aiding an inertial-only navigation system to achieve many tactical strikes. On 14 flight test runs lasting 60 seconds each, the average positional error was 166 feet at endgame, compared with 411 feet achieved by a free-inertial system for a 60% improvement.

Acknowledgements

This thesis is dedicated to my beautiful wife Jennifer. Her incredible support of my interests, passions, and varied crazy ideas speak love more clearly than anything else I could know. I also would like to thank my parents, Willis and Elaine Beich, for providing me with the tools, resources, and love to always do the things I have wanted. Thanks to my daughter, Esme, for reminding me of what is truly important in the midst of work.

I would like to express appreciation to my thesis advisor, Lt Col Michael Veth, for his mentorship and highly-knowledgeable guidance through this research. Dr. John Raquet, Dr. Meir Pachter, and Dr. Richard Martin also provided me with a great professional engineering education, thesis guidance, and inspiration for why this profession matters. I would also like to thank my classmates in USAF Test Pilot School Class 10α, particularly the members of the Project Shuttermatch Test Management Project team (LCDR Brett "Pugs" Pugsley, Maj Kevin "Sonar" Hall, Capt Mark "Magnum" Hanson, Capt J. David "Lapps" Slack, and Capt Nathan "Wheels" Taylor) for their dedication and incredible effort in planning, collecting, analyzing, and packaging the flight data for this research.

I would also like to express thank-yous to the Bomber Barons of the 23d Bomb Squadron and the mighty Pirates of the 31st Test and Evaluation Squadron. Great friends make good jobs even better.

Finally, I thank Christ for giving my life purpose, allowing me to pursue what He has given me passion for, and providing me with far more than I could ever deserve.

Jonathan W. Beich

Table of Contents

List of Figures

List of Tables

List of Symbols

Symbol		Page
χ	Sigma point collection	13
$\mathbf{p}_0^e$	Earth-fixed navigation frame origin vector	18
ψ	Yaw	19
θ	Pitch	19
ϕ	Roll	19
$\mathbf{q}_e^n$	e-frame to n-frame quaternion rotation vector	20
μ	Magnitude of quaternion rotation vector	21
$\mathbf{q}_a^{b*}$	Quaternion complex conjugate	21
$\mathbf{\Omega}_{ne}^e$	e-frame to n-frame skew-symmetric angular rate matrix	22
ω_{nb}^b	b-frame to n-frame angular rotation vector	23
$\mathbf{F}(\mathrm{t})$	State dynamics matrix	26
$\mathbf{x}(\mathrm{t})$	State vector	26
$\mathbf{B}(\mathrm{t})$	Input influence matrix	26
$\mathbf{u}(\mathbf{t})$	Deterministic input vector	26
$\mathbf{G}(\mathrm{t})$	Noise-influence matrix	26
$\mathbf{w}(\mathrm{t})$	Process noise vector	26
$\mathbf{x}_0$	Initial condition state vector	26
$\Phi(t, t_0)$	State transition matrix	26
$\mathbf{p}^n$	Position vector in n-frame	27
$\mathbf{v}^n$	Velocity vector in n-frame	27
$\mathbf{\Theta}^n$	Attitude vector in n-frame	27
$\mathbf{a}^b$	Accelerometer bias in b-frame	27
$\mathbf{b}^b$	Gyro bias in b-frame	27
ψ	Attitude error vector	28
$\mathbf{f}^b$	Specific force in b-frame measured by accelerometers	29

Symbol		Page
Δv^b	Delta-v; accelerometer measurments minus gravity	29
$\Delta\theta^b_{ib}$	Delta-theta; angular rates measured by gyros	29
ω^b_{ib}	Instantaneous angular rate of b-frame w.r.t. i-frame	29
dt	Discrete sample time	29
$\mathbf{g}$	Gravity vector	29
ω^n_{ie}	Sidereal rate rotation vector	32
ω^n_{en}	Transport rate	32
R_0	Radius of the Earth	32
h	Above Ground Level (AGL) altitude	32
$\mathbf{w}^b_a$	Additive accelerometer noise	33
$\mathbf{w}^b_b$	Additive gyro noise	33
$\mathbf{w}^b_{a_{bias}}$	Additive accelerometer bias noise	33
$\mathbf{w}^b_{b_{bias}}$	Additive gyro bias noise	33
$\mathbf{z}_k$	Measurement vector	36
$\mathbf{H}_k$	Measurement observation matrix	36
$\mathbf{v}_k$	Measurement noise vector	36
$\mathbf{K}_k$	Kalman gain	37
$\mathbf{R}_k$	Measurement noise matrix	38
$\hat{\mathbf{x}}$	Mean state vector estimate	39
$\mathbf{P}_{xx}$	State covariance matrix	39
L	Number of states	39
λ	Unscented transform scaling parameter	39
α	Unscented transform function spread parameter	39
κ	Unscented transform secondary tuning parameter	39
$\hat{\chi}^-_{i,k+1}$	*a priori* sigma point state estimate	41
$\mathbf{P}^-_{xx,k+1}$	*a priori* sigma point state estimate	41
$\mathcal{Z}^-_{i,k}$	Sigma point predicted measurement	41
$\mathbf{s}^c$	World projection vector (3D);camera to object	42

Symbol		Page
$\mathbf{s}^{proj}$	Image plane projection vector (2D)	42
W	Camera sensor width	43
H	Camera sensor height	43
$\mathbf{T}_c^{pix}$	Camera-to-target vector to pixel coordinate transformation	43
M	Vertical pixel dimension	44
N	Horizontal pixel dimension	44
ϕ_e	Geodetic latitude	59
λ	Geodetic longitude	59

List of Abbreviations

Abbreviation		Page
SPR	Statistical Predictive Rendering	1
AFIT	Air Force Institute of Technology	1
VANSPR	Vision-Aided Navigation using SPR	1
TMP	Test Management Project	1
TPS	Test Pilot School	1
ANT	Advanced Navigation Technology	1
JDAM	Joint Direct Attack Munitions	2
INS	Inertial Navigation System	2
GPS	Global Positioning System	2
LGB	Laser-Guided Bomb	2
CEP	Circle Error Probable	2
COTS	Commercial Off the Shelf	6
TSPI	Time Space Position Inertial	6
gyro	gyroscope	6
SIFT®	Scale-Invariant Feature Transform	7
DCM	Direction Cosine Matrix	13
UT	Unscented Transform	13
ECEF	Earth Centered Earth Fixed	15
VRML	Virtual Reality Markup Language	16
WGS-84	World Geodetic System 1984	23
DoD	Department of Defense	24
EGM96	Earth Gravitational Model 1996	24
LTI	Linear Time-Invariant	25
WGN	White Gaussian Noise	26
AGL	Above Ground Level	32

Abbreviation		Page
FOGM	First Order Gauss-Markov	33
EKF	Extended Kalman Filter	38
UKF	Unscented Kalman Filter	38
PDF	Probability Density Function	38
PF	Particle Filter	38
SURF	Speeded Up Robust Features	47
SSD	Sum of Squared Differences	49
SAD	Sum of Absolute Differences	49
NCC	Normalized Cross Correlation	49
ZSAD	Zero Mean SAD	50
LSAD	Locally Scaled SAD	50
FFT	Fast Fourier Transform	51
LMedS	Least Median of Squares Regression	51
RANSAC	Random Sample Consensus	51
DOF	Degrees of Freedom	76
TCG	Time Code Generator	88
MFD	Multi-Function Display	88
MP	Megapixel	88
fps	Frames per Second	91
MHz	Megahertz	93
GB	Gigabytes	93
RAM	Random Access Memory	93
GHz	Gigahertz	93
TB	Terabyte	93
TIFF	Tagged Image File Format	94
GAINR	GPS-Aided Inertial Navigation Reference Unit	94
IMU	Inertial Measurement Unit	94
PCMCIA	Personal Computer Memory Card International Association	94

Abbreviation		Page
CAD	Computer Aided Design	96
PIRA	Precision Impact Range Area	96
AFB	Air Force Base	96
AFFTC	Air Force Flight Test Center	96
SAEC	Solar Active Edge Corner	97
DOE	Design of Experiments	97
KIAS	Knots Indicated Air Speed	100
nm	Nautical Miles	104
DTED	Digital Terrain Elevation Data	165
LADAR	Laser Detection and Ranging	166
GPU	Graphics Processing Unit	166
AI	Artificial Intelligence	167
SoS	System of System	167

Vision-Aided Autonomous Precision Weapon Terminal Guidance Using a Tightly-Coupled INS and Predictive Rendering Techniques

I. Introduction

This thesis describes the development of a vision-based navigation solution that compares images captured by an on-board (mounted on aircraft for the purposes of flight test) camera system with synthetically-generated images to determine relative position. The underlying technique, Statistical Predictive Rendering (SPR), was used previously in an Air Force Institute of Technology (AFIT) project to improve upon relative navigation during aerial refueling [37]. The work presented here applies the concept via a new test algorithm coined Vision-Aided Navigation using Statistical Predictive Rendering (VANSPR) to more complex and ground-based object environments. In-flight data for this research was collected during a Test Management Project (TMP), named Project Shuttermatch [4], during the author's education at the Air Force Test Pilot School (TPS) as part of the Joint AFIT/TPS program. The emphasis of this research was terminal weapon navigation in a GPS-denied environment, a primary focus of AFIT's Advanced Navigation Technology (ANT) Center. The VANSPR algorithm has potential to also be utilized in a wide variety of other navigation applications.

1.1 Motivation

Modern air warfare takes precision strike as a precondition for operations. This fact has been most visibly evidenced over the past decade in the Afghanistan and Iraq wars where physical proximity between strategic targets of interest and civilian property and lives may be separated by mere feet. The political and moral implications of errant weapons can be devastating; hence, the requirement for precision is obvious, as

expressed in the official US Air Force Doctrine-1 Publication: "... with the advent of precision weaponry, the US is capable of carefully regulating the destructive effects of [strategic attack] thereby minimizing collateral damage. This capability enables the US to use these coercive mechanisms in a way that complies with the laws of armed conflict" [9]. In general, the more *precise* a weapon can be, the *better* the weapon is.

1.1.1 Precision and Near-Precision Weapons. Because this research ultimately seeks to enhance military weapon utility, the state-of-the-art in precision weapons must be discussed briefly. Air-to-ground weapons with some type of active targeting can be broadly classified as near-precision or precision. The term *near-precision* refers to weapons such as the Joint Direct Attach Munitions (JDAM) that navigate to a programmed coordinate location, but have no real perception of the environment around them. JDAMs navigate through use of an Inertial Navigation System (INS) and a Global Positioning System (GPS) receiver. While GPS is heavily relied upon for the best possible solution, a less accurate INS-only solution may also be obtained by mathematical integration of sensed accelerations and angular rates. *Precision* weapons, like the Laser-Guided Bomb (LGB), have a seeker head that detects a laser beam spot on a target that the operator actively directs, and navigates to the laser spot location. As the names imply, precision weapons generally perform more accurately than near-precision weapons. The unclassified advertised JDAM Circle Error Probable (CEP) is 13 meters using GPS and 30 meters without GPS guidance [33]. The CEP statistics for LGBs are classified.

Both of these weapon types have clear disadvantages. The JDAM navigation solution is limited by the accuracy of the coordinates that it is given. In a stressful, quickly-evolving target environment, erroneous coordinates have been programmed into weapons with grave consequences. This problem is partially mitigated with new technologies that allow combat aircrew to receive coordinates via datalink and send them to a bomb without additional hand-copying and re-typing. A disadvantage of the LGB is the requirement for a pilot or other asset to remain in the target environment

during the 30-90 second time-of-flight. This required loiter time to illuminate a target may cause unnecessary exposure to threats and/or prevent prosecution of a subsequent high-value, time-sensitive target.

Recent work has been made to create laser JDAMs which add a seeker head to a conventional JDAMs to give them the additional abilities of an LGB. While increasing the flexibility, and hence operational value, of the weapon, it does not overcome or circumvent the inherent problems of each weapon type. A laser JDAM may be able to precisely strike a moving tank, but the ability is lost if a cloud layer is present between the target and strike aircraft. Likewise, the JDAM functionality of the weapon would make it useable to drop through the cloud layer, but the benefit of a precision weapon would be lost.

1.1.2 Dependence on the Global Positioning System. Although used perhaps most conspicuously by JDAMs, the use of GPS for precision navigation is widespread. The current reliance on GPS in military systems, including weaponry, has lead to a complacency concerning its future availability. In other words, users may not know what they have until it is taken from them. Contemporary conflicts have shown the vulnerabilities and potential exploitation of the GPS system on a relatively small scale, but it is almost certain that any future large-scale engagement with a technologically adept adversary would involve an effort to deny precision navigation capability by any possible means. This threat has led to an intensifying science and engineering effort, expressed by one contemporary professional magazine as: "The [Air Force] must focus on developing technologies for air and space systems that enable it to maintain air dominance in hostile territory. Three research areas are deemed particularly important in this regard: precise navigation and timing in GPS-denied environments, electromagnetic-spectrum warfare, and cyber resilience" [8]

In the last several years, much work has been done to find a reliable alternative to GPS aiding of navigation systems. While GPS is, and will likely remain for many years, appropriate for many commercial and military uses, vulnerabilities exist

which motivate the need for back-up and replacement systems and methods. Some reasons for GPS nonavailability may be inherent: indoor navigation, use at extreme latitudes, or even extraterrestrial operations beyond the current GPS constellation. Nonavailability may also be caused by unintentional sources such as signal blockage from terrain in mountainous areas, buildings in urban environments, foliage cover in forest or jungle, or atmospheric interference. A third source is malevolent signal interference, which may or may not be recognized, depending on the form it takes. This type of interference may cause outright disruption in service, increased error, or clever deception resulting in high confidence in incorrect position. Such sabotage may target either military or commercial assets to gain tactical or economic advantage [5].

The fact that a GPS-receiver works by performing relatively simple algorithms on the signals it receives from a high-tech multi-billion dollar satellite constellation lends to its small size and expense. The tradeoff, however, is that it must rely on outside signals to perform. Solutions that feature an INS are self-contained but require updates from some additional sensor or sensors for accuracy over an extended period of time. Combined INS/GPS systems, as previously mentioned regarding JDAM, combine the accuracy of GPS with the statistical rigor of a Kalman-filtered INS. With a lapse in GPS information, the INS will continued to navigate, albeit in a degraded state. It is desired, therefore, to have self-contained passive sensors that can be integrated with the INS to improve its navigation accuracy and reliability. Such sensors, like cameras, are mostly resilient to electronic jamming and other types of spoofing, and have been shown to significantly improve an INS-only system [15] [35] [23].

1.1.3 Vision-Aided Navigation. The ideal model of a vision-based system already exists in most humans as eyes, brain, and supporting subsystems. While it might be thought that machines could more easily conquer any task more quickly and efficiently than organic life, as has been shown with robots that perform tasks from automobile manufacturing to playing chess, more cognitive tasks pose an extremely difficult problem. This is in part due to the large amounts of contextual and intuitive

information that humans and animals unconsciously apply to the raw data that their eyes see. Computers simply lack the natural ability to fill in gaps in information - unless they are somehow told to do so by their programming. Therein lies the challenge of computer vision applications. The problem is somewhat simplified in the application presented in this research in that while the perspective of the environment is changing in time, the objects in the environment are assumed to be stationary.

Computer vision always implies some type of comparison, whether it is detecting change or movement from one frame to the next or attempting to identify an individual object in a scene by comparing the image with a database of what an object should or may look like. If information (i.e., position and attitude) can be extracted from these differences that reveal how the scene has changed, then the surmised errors can be removed from the navigation solution.

1.2 Problem Statement

The impetus for this research is to contribute toward evolving *smart weapons* into *brilliant weapons* by developing a navigation algorithm that maintains the advantages of precise weapons (what part of the tank do I want to hit?) *and* non-precision weapons (can I drop even if I can't see the target?) and can be used in a GPS-denied environment. Past vision-based efforts have primarily treated the navigation problem as flat and two-dimensional [23, 25, 35, 36] and generally offered no way to improve performance when it is most desired. This research examines the special case and consequent issues of considering the three-dimensional aspect of the environment and, for the case of a precision weapon, greater potential accuracy as the system nears its target. To accomplish this, a unique algorithm is developed that compares images taken by the on-board camera system with predictively-rendered views of a three-dimensional target environment model to determine navigation error. Since the models can be constructed to a high degree of fidelity and resolution, navigation solution refinement can be increased as the observer gets closer to three-dimensional objects in the environment. Simulation will be used to develop the Kalman filter

mechanization and flight test will be used to collect images, truth data, and raw inertial data to practically test the algorithm in a "real world" environment. The research in this thesis specifically addresses the relative navigation problem of a tightly-coupled INS/vision system in a three-dimensional, GPS-denied environment.

1.3 Research Goals and Contributions

In addition to the overall objective of contributing towards a greater weapon capability as stated in the section above, several specific research goals and contributions to the navigation field are set. These include improving the navigation solution over an INS-only system, maintaining the ability to track objects throughout greatly changing viewpoint geometry, lowering the computation burden required for feature tracking in changing geometry, demonstrating the use of inexpensive Commercial Off the Shelf (COTS) software for complex targeting, and improving the navigation solution as the target is neared.

1.3.1 Navigation Solution Improvement over INS Alone. Flight test data included raw INS for constructing a navigation system as well as post-flight processed Time-Space-Position-Inertial (TSPI) to be used as a truth source. The raw INS data, consisting simply of the raw three-axis accelerometer and rate gyroscope gyro) measurements (six measurements per sample time), were run through the appropriate mechanization equations to construct an unaided flight path trajectory. A more detailed discussion of the mechanization equations will be given in Chapter 2. The primary specific objective of the VANSPR algorithm is to show performance improvement over the INS-only solution. A GPS-aided "real-time" solution was also available as a means of algorithm comparison, which was practically indistinguishable from the TSPI data without close examination. While this trajectory offers another means of comparison, the VANSPR algorithm performance is not expected to be as good as GPS-aided or TSPI solutions.

1.3.2 Track Objects Throughout Greatly Changing Viewpoint Geometry. The task of comprehending that the same object is being observed after the viewing aspect has been substantially changed is a very challenging problem in the field of computer vision. Considering the images shown in Figure 1.1, it would be unlikely that a feature-matching algorithm would be able to correctly determine that the starred feature points in one image matched their counterparts in the other image. Some algorithms like the Scale-Invariant Feature Transform® (SIFT®) [19] can withstand some affine perspective change, but will eventually break down. This can be a problem when attempting to compare subsequent images taken from a camera in a vision-aided navigation system.

An advantage of comparing an actual image to a synthetically generated image, as used in the VANSPR algorithm, is that the two images should be similar enough that a large perspective change would not be present between the two. A feature-matching algorithm would simply see two very similar images. The assumption is that a *particular* feature point is not important, but only the presence of finding *some* feature points in the image pair for the purposes of extracting information from the differences.

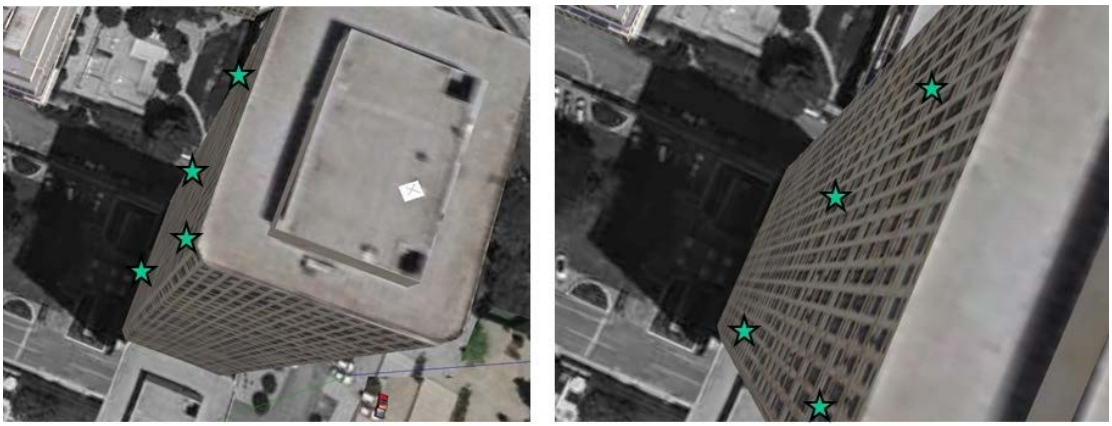


Figure 1.1: Feature-point matches on same building with significantly changed perspective. Even highly-robust feature correlation algorithms have difficulty finding these matches.

1.3.3 Lower Computational Burden for Three-dimensional Processing. Computationally robust, and therefore computationally expensive, feature-matching algorithms such as SIFT® are partially invariant to affine perspective changes, making it an ideal tool for cases where that is expected. However, when a pair of images being compared does not have such perspective difference, a faster, less burdensome algorithm can be applied.

1.3.4 Commercial Off-The-Shelf (COTS) Software for Complex Targeting. Three-dimensional models used in the research were built using the Google Sketchup Pro® software application, readily available for download for anyone with an internet connection and the minimum hardware and operating system requirements. The primary advantages to using this software package is that it is user-friendly, has a large user community driving frequent updates and bug fixes, and can be quickly obtained on new mission planning computers. Target environment design can even be accomplished on the free version of the software, Google Sketchup®. The Pro version is required to export the models to MATLAB®, as will be explained in greater detail later.

1.3.5 Improving Solution as System Nears Target. As with most other vision-aided navigation techniques where it is desired to navigate to an object, the solution accuracy increases as the target approaches and grows larger in the image. The scale of error correction becomes *refinement* if the predictive algorithm has sufficiently matched the real world up to that point. This ability creates a type of *self-deconfliction* within the weapon's navigation system where decisions based on geometry or recognizable features can be exploited even if the precise desired impact point cannot be seen. If the camera and algorithm could be operated real-time and at a high enough rate, and if the three-dimensional model was built to a high-enough degree of accuracy, the last image processed could theoretically be a correction of centimeters. The resolution of detail seen in the three-dimensional target models is limited by the resolution of the images used for surface texture-mapping.

1.4 Scope and Assumptions

The use of SPR to solve the position problem is relatively new to the field of navigation engineering; therefore, there is much to evaluate and discuss. The research could eventually extend to indoor navigation and robotic/vehicular navigation in cities ("urban jungles") and incorporate a variety of additional sensors. The focus of SPR in the context of this thesis is autonomous navigation for the purpose of terminal weapon guidance, mechanized in a tightly-coupled INS and Kalman filter system.

Figure 1.2 depicts a conceptual overview of how the VANSPR algorithm works. Viewpoints of a constructed "world model" are generated by *a priori* navigation state estimates and compared with actual camera images. Special image assessment techniques are used to extract error directions and magnitudes which can then be used to correct the navigation solution to provide a more accurate *a posteriori* estimate.

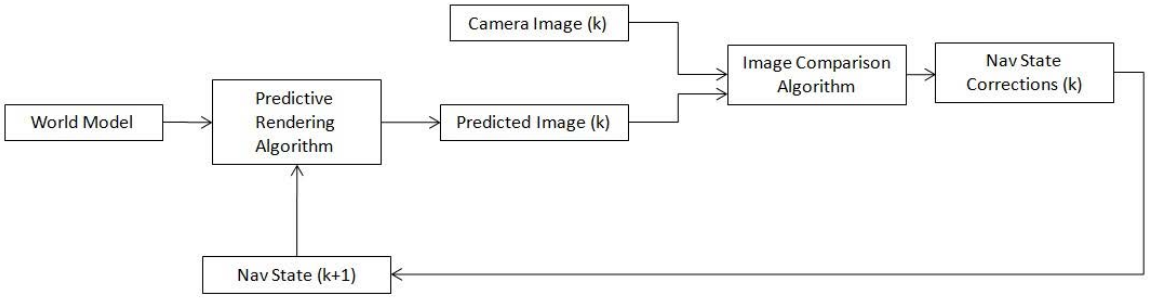


Figure 1.2: Conceptual overview of the VANSPR algorithm. Camera images are compared with computer-generated images of what the scene is expected to look like. Differences in these images are used to determine errors which may then be removed from the navigation estimate.

Assumptions of this research include:

- The VANSPR algorithm solves only the navigation aspect of the problem. The weapon flight control system is assumed to be pre-existing.
- The VANSPR algorithm is online (causal). However, this proof-of-concept is post-processed; a real-time version is left for future work.

- All objects are stationary relative to the navigation frame.
- Target environment locations are accurate to 0.025 meter [14].
- GPS is not available to the VANSPR test algorithm.
- No lasing, range-finding, or any other kind of update besides camera images will be available to algorithm.
- An initial handoff of navigation states and covariance is available from TSPI data at "release" time.
- INS attitude errors are assumed to be negligible over the weapon time of flight.
- The autogain software settings of the test camera provide the optimal picture and will not be changed.

1.5 Thesis Overview

This thesis is divided into five chapters to present the background, algorithm development, laboratory and flight test procedures, experimental results, and offer conclusions and recommendations. Chapter 2 builds the required knowledge to understand the problem in a mathematical sense to include basic notation, reference frames and transformations, Earth modeling and mapping, system modeling, INS concepts, digital imaging, methods of image comparison, Kalman filtering, three-dimensional modeling, and previous work.

Chapter 3 offers a detailed description of the VANSPR algorithm development from the establishment of the system model to time-alignment of the flight test data. The mathematical development of the Kalman filter that was used will be given, along with a description of what image comparison and processing techniques were used.

Chapter 4 details the hands-on portion of the research through simulation and flight test. Miniature simulations of various pieces of the VANSPR algorithm were conducted to ensure they would work in the final product. A large amount of varied data over the course of eight test flights was collected to provide meaningful points

of comparison and insure against transient problems. The research will consider only the relevant weapon profile maneuvers that were flown.

Chapter 5 Presents the experimental results of the flight tests. This includes verification of the ability to render synthetic images that closely resemble actual camera images when provided with truth data, verification of various image comparison methods with typical flight test data, and exploration of modifications to data processing.

Finally, Chapter 6 will offers conclusions to the examined data and makes recommendations for future work. While this research attempts to evaluate the utility of using SPR to strike ground targets with a guided weapon, more questions will inevitably be formed from the investigation and present opportunities for further exploration.

II. Background

This chapter provides the conceptual and mathematical background necessary for the upcoming VANSPR algorithm development discussion. A basis of mathematical notation will be presented first, followed by a review of reference frames, how reference frames may be transformed, and the role of Earth modeling and mapping. Next, system modeling and INS concepts will be discussed in a more mathematics-intensive manner. The vision-based aspect of the research will then be examined in the sections on digital imaging and image comparison techniques. A review of Kalman filtering, to include the unscented Kalman filter, and vision-aiding will be discussed, followed by how the target environments were three-dimensionally modeled. Finally, a brief review of relevant previous research will be conducted.

2.1 Mathematical Notation

This thesis uses the following mathematical notation:

- **Scalars** : upper or lower case letters in italic type, (e.g., a or A).
- **Vectors** : lower case letters in bold type, (e.g., $\mathbf{x}$). Vectors are assumed to be column vectors unless otherwise noted, comprised of scalar elements where x_i is the i^{th} scalar element of $\mathbf{x}$.
- **Unit Vectors** : denoted with a *check* symbol above the vector letter and defined by the two norm, (e.g., $\|\check{\mathbf{x}}\| = 1$).
- **Matrices** : upper case letters in bold type, (e.g., $\mathbf{X}$). The element X_{ij} is the scalar component of $\mathbf{X}$ from the i^{th} row and j^{th} column.
- **Transpose** : denoted with a superscript $\mathbf{T}$, (e.g., $\mathbf{x}^{\mathbf{T}}$). The transpose can be applied to a vector or matrix.
- **Estimated Variables** : denoted with a *hat* symbol above the variable name, (e.g., $\hat{x}$). This is an estimate of a random variable.
- **Computed Variables** : denoted with a *tilde* symbol above the variable name, (e.g., $\tilde{x}$). This is a computed variable and hence corrupted by noise.

- **Measured Variables** : denoted with a *bar* symbol above the variable name, (e.g., $\bar{\mathbf{x}}$). This is a measured variable and hence corrupted by noise.
- **Homogeneous Coordinates** : denoted by an *underline* below the variable name, (e.g., $\underline{\mathbf{x}}$). This is a two-dimensional coordinate represented as a 3×1 matrix for mathematical reasons; the last element is a 1.
- **Direction Cosine Matrices** (**DCM** :) uppercase letter **C** denoted with subscript and superscript letters, (e.g., $\mathbf{C}_a^b$). This is a 3×3 matrix that transforms a 3×1 vector in the a coordinate system into the b coordinate system.
- **Quaternions** : lowercase letter q denoted with subscript and superscript letters, (e.g., $\mathbf{q}_a^b$). This is a 4×1 vector that transforms a 3×1 vector in the a coordinate system into the b coordinate system.
- **Reference Frame** : denoted with a superscript on the vector name indicating what coordinate system it is represented in, (e.g., $\mathbf{x}^a$).
- **Relative Position or Motion** : denoted with a two-lettered subscript representing the motion of one reference frame with respect to another reference frame, and often with a superscript indicating what reference frame it is represented in, (e.g., ω_{ab}^c means it is a rotation rate vector of the b frame relative to the a frame, represented in the c frame).
- **Time Derivatives** : denoted with dots above the vector, DCM, or quaternion, (e.g., $\dot{\mathbf{r}}$(t), $\ddot{\mathbf{r}}$(t), $\dot{\mathbf{C}}_e^n$, $\dot{\mathbf{q}}_e^n$). One dot indicates a first derivative, a second dot indicates a second derivative, etc.
- **Sigma Points** : Greek letter *chi* (χ) represents an L×(2L+1) matrix of values generated by the Unscented Transform (UT), where L is the number of states. The subscript denotes the matrix that has undergone the UT, (e.g., $\chi_{\mathbf{a}_k^b}$ is the transformed version of $\mathbf{a}_k^b$).

2.2 Reference Frames

Bodies of interest require a concise mathematical and conceptual description to convey meaning of position and motion. This is accomplished through the use of reference frames. Reference frames may be classified as *inertial*, meaning the frame itself is not accelerating, or *noninertial*, meaning the frame is accelerating. The acceleration of the noninertial frame may be translational, rotational, or a combination of both. Rotating reference frames are considered noninertial even when no actual rotational acceleration exists *within* the frame; linear accelerations are induced in frames that have a constant rotational velocity, hence they are noninertial frames. In an inertial reference frame, the equations of classical mechanics can be used without modification; however, in a noninertial reference frame, equations of motion must be modified to account for "fictitious force" induced by the frame's acceleration.

Reference frames may be defined in various geometric coordinate systems such as rectangular (Cartesian), spherical, and cylindrical. In this thesis, right-handed Cartesian coordinate frames with mutually perpendicular (orthonormal) x, y, and z axes will be used primarily. A brief description of the reference frames that will be used or needed in this research will be discussed next.

2.2.1 True Inertial Frame (I-frame). The only truly inertial reference frame in the universe is the I-frame. In this reference frame, motion equations can be directly applied without compensating for fictitious forces of noninertial frames. Since there are an infinite number of I-frames, no origin is defined (or needed).

2.2.2 Earth-Centered Inertial Frame (i-frame). The origin of the i-frame is at the center of the Earth, with two of its axes defined by "fixed" stars. The x-axis is defined in the vernal equinox direction. By strict definition, the vernal equinox direction is found by drawing a line from the Earth to the Sun on the first day of spring. When this was determined several thousand years ago, this line also pointed to the first star in the Constellation Aries [27]. Although no longer strictly true (the

line now points through the Pisces constellation), the x-axis is often defined in the literature as pointing to the First Point in Aries. The z-axis points at the star Polaris (North Star), and the y-axis is orthogonal to the x and z axes. Since the Earth rotates but the i-frame does not, a stationary point in the i-frame would move across the Earth in time.

2.2.3 Earth-Centered Earth-Fixed Frame (e-frame). Also referred to as the ECEF-frame, the e-frame shares the center of the Earth as its origin with the i-frame. However, unlike the non-rotating i-frame, the e-frame rotates with the Earth. A stationary point, with respect to the surface of the Earth, defined in the e-frame has constant geodetic coordinates.

2.2.4 Navigation Frame (n-frame). The origin of the n-frame is at the navigation system (INS in an aircraft) and has axes defined by true north, east, and down. Coordinate systems with this orientation are commonly referred to as *NED coordinate systems.* "Down" is defined by the direction of the gravity vector. Because the origin of the navigation system is within the aircraft, the frame moves with the aircraft position but is always oriented the same regardless of aircraft attitude. Navigation frames (n-frames) differ for different bodies of interest; only one body, the aircraft, is considered here.

2.2.5 Earth-Fixed Navigation Frame (n'-frame). An additional NED coordinate system will be used which has the same axis orientation previously described for the n-frame, but that does not move with the aircraft. The n'-frame will be used to describe movement relative to aircraft position at a specific point in time. In practice, the n'-frame origin will be declared collocated with the n-frame origin at the beginning of a data collection run but remain stationary with respect to the Earth to provide a convenient means of observing relative movement over the course of the run. Figure 2.1 shows the Earth-centered inertial, Earth-centered Earth-fixed, and navigation frames together.

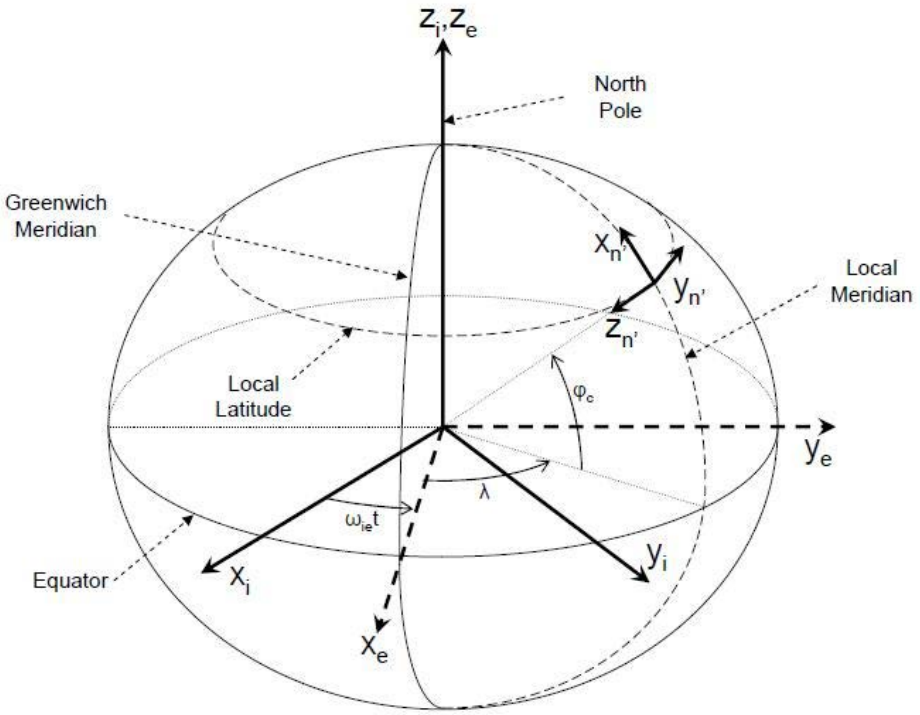


Figure 2.1: Earth-centered inertial, Earth-centered Earth-fixed, and navigation frames [35] . The symbols λ, ϕ_e, and $\omega_{ie}t$ refer to longitude, latitude, and Earth rotation angle, respectively.

2.2.6 Body Frame (b-frame). The origin of the b-frame is collocated with the n-frame; however, the axes rotate with respect to aircraft attitude. The x, y, and z axes are defined in the direction of the aircraft's nose, right wing, and bottom of the aircraft respectively. A C-12C aircraft, used during the data collection test flights, is shown in Figure 2.2 with the b-frame axes superimposed.

2.2.7 Camera Frame (c-frame). The c-frame's origin is the camera's optical center. The z-axis points out the camera's lens. The x-axis points up and the y-axis points out the right side of the camera as shown in Figure 2.3. The plane created by the x and y-axes parallels the focal plane.

2.2.8 VRML Frame (v-frame). Target models constructed in the Virtual Reality Markup Language (VRML) world comply with yet another coordinate system,

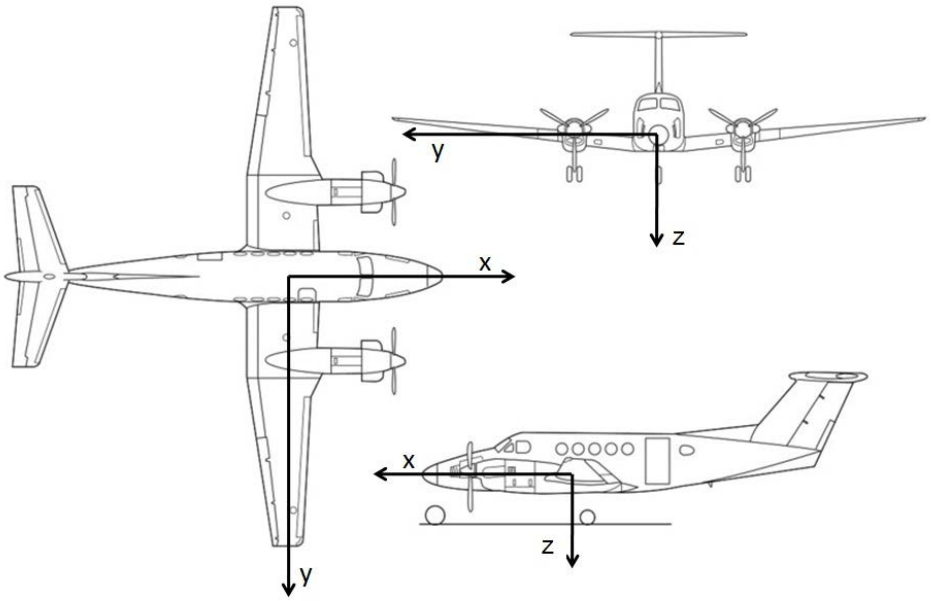


Figure 2.2: Body frame axes defined on a C-12C aircraft. This was the aircraft used for the data collection test flights.

Figure 2.3: Camera axes superimposed on Prosilica 4900 camera. The camera installed on the aircraft was rotated 90° counter-clockwise to optimize the view vertically.

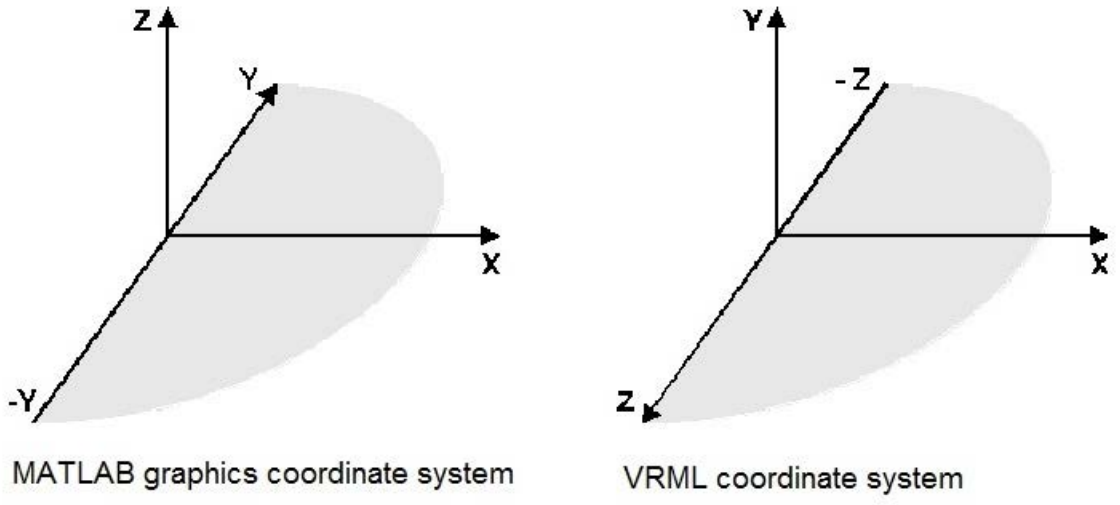


Figure 2.4: MATLAB® standard graphics and VRML synthetic world coordinate systems [30]. These differences required rotational transformations to correctly display the computer-generated models with respect to aircraft estimated position and attitude.

different from the MATLAB® standard for three-dimensional coordinates shown in Figure 2.4. Knowledge of both of these frames is required since VRML objects are manipulated through MATLAB®.

2.3 Reference Frame Transformations

Reference frame transformations consist of two separate steps: translation and rotation. Translation simply moves the origin of one frame to the next by means of an additive vector:

$$\mathbf{p}^n = \mathbf{p}^e - \mathbf{p}_0^e, \tag{2.1}$$

where $\mathbf{p}_0^e$ is the three-dimensional position vector from the origin of the e-frame to the origin of the n-frame, expressed in the e-frame.

While the reference frame coordinates locate the position of a body in space, another mathematical representation must be used to describes the body's attitude

with respect to the reference frame. The three most common attitude representations are Euler angles, direction cosine matrices, and quaternions.

2.3.1 Euler Angles. Euler angles consist of the parameters yaw (ψ), pitch (θ), and roll (ϕ) and are typically defined by the three two-dimensional planes in the b-frame of an aircraft relative to the three two-dimensional planes of the collocated n-frame. Unlike other coordinate system angle transformations that can be done in one step, the conversion of Euler angles to another frame is done one angle at a time. Any order can be used, as long as it is applied consistently throughout all mathematical manipulation. The order most commonly used is yaw, then pitch, and then roll. An attempt to make a transformation with the same angles but rotated in a different order will yield an incorrect result. Yaw may also be referred to as heading. Positive Euler angles indicate the rotation is in accordance with the right-hand rule for the axis it rotates about. The methods described in the next two subsections are most commonly used for angular coordinate system transformations, while the Euler angles are typically calculated and displayed to the pilot.

2.3.2 Direction Cosine Matrices (DCMs). A direction cosine matrix (DCM) is a 3×3 matrix with columns that represent unit vectors in one reference frame projected along the axis of another reference frame. Unlike the Euler angle representation, no regard for rotation order needs to be made. A complete mathematical explanation can be found in Titterton [31]. Once the appropriate DCM is constructed for transformation between reference frames, a vector in the original frame is pre-multiplied by the DCM to realize the vector in the new frame:

$$\mathbf{v}^n = \mathbf{C}_e^n \mathbf{v}^e \tag{2.2}$$

For transformations between several reference frames, successive DCMs can be pre-multiplied for each single reference frame conversion. Careful attention must be paid to the order of matrix multiplication:

$$\mathbf{p}^b = \mathbf{C}_n^b \mathbf{C}_e^n \mathbf{C}_i^e \mathbf{p}^i \tag{2.3}$$

Consideration of some special properties of the DCM when used to transform right-hand Cartesian coordinates should be made to ensure that they are maintained when the vector mathematics are computed in a digital computer. These properties are:

$$Det(\mathbf{C}_a^b) \equiv |\mathbf{C}_a^b| = 1 \tag{2.4}$$

$$\mathbf{C}_b^a = (\mathbf{C}_a^b)^{-1} = (\mathbf{C}_a^b)^{\mathbf{T}} \tag{2.5}$$

$$\mathbf{C}_a^c = \mathbf{C}_b^c \mathbf{C}_a^b \tag{2.6}$$

2.3.3 Quaternions. An elegant, albeit slightly more difficult to implement, mathematical method of reference frame rotation is the quaternion. Quaternions are 4×1 matrices that define a single vector about which one reference frame may be rotated by a given angle to achieve a new reference frame. A quaternion has one real component and three imaginary components:

$$\mathbf{q}_e^n = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k} \tag{2.7}$$

where $\mathbf{q}_e^n$ is a reference frame rotation from the e-frame to n-frame, a is the real component, and b, c, and d are coefficients of the imaginary $\mathbf{i}$, $\mathbf{j}$, and $\mathbf{k}$ components. The quaternion can also be expressed as

$$\mathbf{q}_e^n = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} cos\ (\mu/2) \\ (\mu_x/\mu)\ sin\ (\mu/2) \\ (\mu_y/\mu)\ sin\ (\mu/2) \\ (\mu_z/\mu)\ sin\ (\mu/2) \end{bmatrix} \tag{2.8}$$

where μ is the magnitude and μ_x, μ_y, and μ_z are the components of the rotation vector.

In order to apply a quaternion transformation to a vector, the vector's dimension must first be increased to 4×1 with the addition of a zero term to the real component's vector location.

$$\mathbf{r}^a = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \Rightarrow \mathbf{r}^{a'} = \begin{bmatrix} 0 \\ x \\ y \\ z \end{bmatrix} \tag{2.9}$$

The vector can then be transformed with a quaternion by:

$$\mathbf{r}^{b'} = \mathbf{q}_a^b\ \mathbf{r}^{a'}\ \mathbf{q}_a^{b*} \tag{2.10}$$

where $\mathbf{q}_a^{b*}$ denotes the complex conjugate of $\mathbf{q}_a^b$. The desired vector, $\mathbf{r}^b$ can then be extracted by removing the leading zero of the first element:

$$\mathbf{r}^{b'} = \begin{bmatrix} 0 \\ \mathbf{r}^b \end{bmatrix} \tag{2.11}$$

Similar to DCMs, quaternions can also be multiplied together to transform a vector through multiple reference frames. Also like DCMs, careful attention must be paid to the order of multiplication to ensure a correct result.

2.3.4 Propagation of Rotations in Time. If the reference frames of interest continue to change with respect to each other over time, a differential equation must be used to describe the DCM or quaternion. It is assumed that Euler angles will be converted into one of these forms before continuing.

A DCM propagating in time can be described by:

$$\dot{\mathbf{C}}_e^n = \mathbf{C}_e^n \mathbf{\Omega}_{ne}^e \tag{2.12}$$

where $\mathbf{\Omega}_{ne}^e$ is the skew symmetric rotation vector of frame e with respect to frame n, resolved in the e-frame.

The rotation vector ω, expanded as

$$\omega = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{2.13}$$

can be expressed in skew symmetric form as

$$\omega\times = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{2.14}$$

The propagation of a quaternion in time may be expressed as

$$\dot{\mathbf{q}}_b^n = \frac{1}{2}\mathbf{q}_b^n \mathbf{p}_{nb}^b \tag{2.15}$$

where

$$\mathbf{p}_{nb}^b = \begin{bmatrix} 0 & \omega_{nb}^{b\mathbf{T}} \end{bmatrix}^{\mathbf{T}} \tag{2.16}$$

where ω_{nb}^{b} is the angular rotation of the body frame with respect to the navigation frame, expressed in the body frame.

2.3.5 Simultaneous Translation and Rotation. If a position vector needs to be resolved in another reference frame that requires both a translation and rotation transformation, both operations may be expressed in one equation as

$$\mathbf{p}^{b} = \mathbf{C}_{n}^{b}(\mathbf{p}^{e} + \mathbf{p}_{0}^{e}) \tag{2.17}$$

2.4 Earth Modeling and Mapping

2.4.1 Basic Geodesy. Geodesy is the study of the measurement and representation of the Earth. The ties to navigation on and above the Earth are obvious: the more accurate the model, the more accurate the navigation solution. In the case of Air Force interests, the more accurate the model and navigation solution, the more effective the weaponry.

A datum is a collection of reference points used to determine the location of any other points around or on the surface of the Earth. The datum database on any modern commercial hand-held GPS receiver will show that many datums exist to describe a geographic position, with some being useable for only a limited area. The World Geodetic System 1984 (WGS-84) is currently the most widely recognized datum.

The Earth is not a perfectly uniform shape; it is most accurately described in mathematical terms as an oblate spheroid. For the purposes of modeling and navigation, two common models of the Earthare the ellipsoid and the geoid. A datum defines a unique ellipsoidal model of the Earth with minimum error in the area of interest. A geoid is a gravitation equipotential model of the Earth. This means that at any given point on a geoid, the surface is perpendicular to local "down" or the direction a plumb line would point. Because of mass inconsistencies in the Earth and at the surface, local gravity vectors are not equal in magnitude and are not uniformly

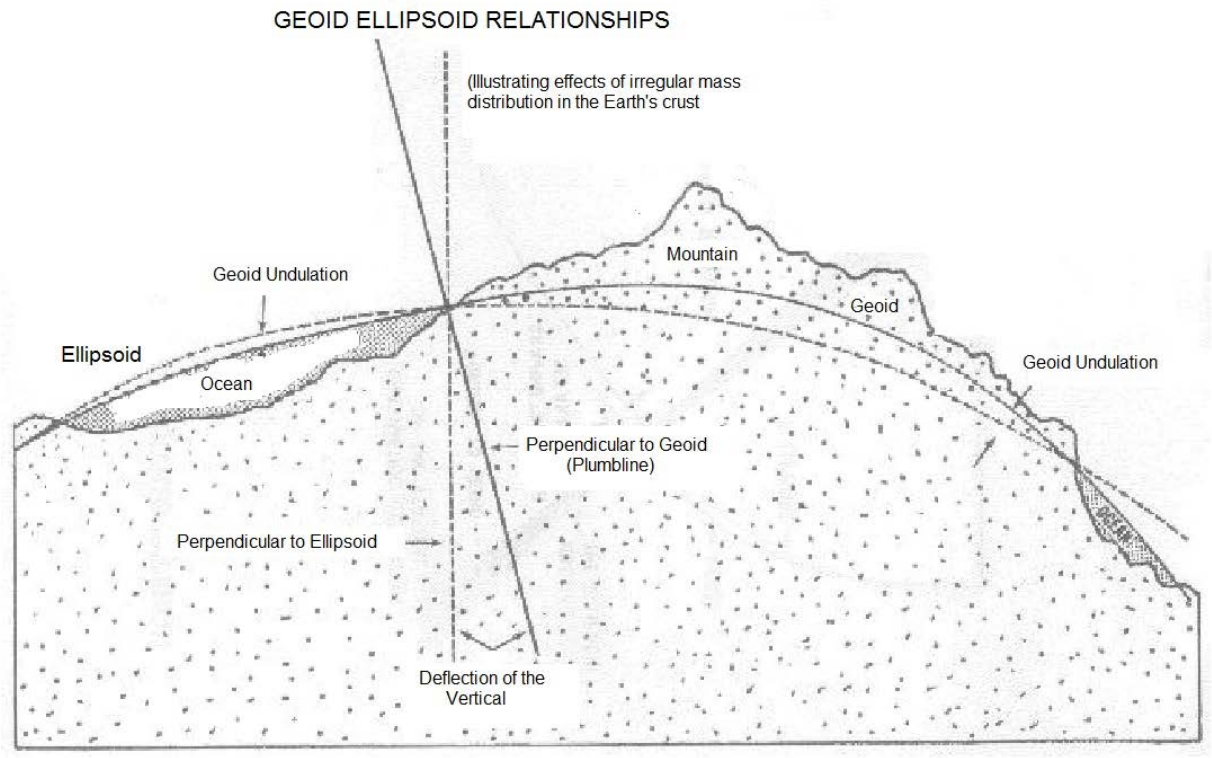


Figure 2.5: Ellipsoid and geoid models compared to the actual surface of the Earth [10]. These three references may be collocated or differ by several hundred feet.

pointed toward the exact center of the Earth. Figure 2.5 demonstrates the differences between a model ellipsoid, a geoid, and the Earth's surface.

2.4.2 WGS-84. WGS-84 provides a common global framework for all geospatial information within the Department of Defense (DoD) and globally for GPS users. WGS-84 provides a comprehensive coordinate system, a reference ellipsoid, and a geoid model. The system has been updated several times since its inception in 1984, with the most significant update for navigation purposes being the Earth Gravitational Model 1996 (EGM96) [22]. The e-frame will be used to represent the WGS-84 ellipsoid as shown in Figure 2.6.

Now that reference systems and the associated mathematics required to transition between them has been discussed, the background discussion will continue with

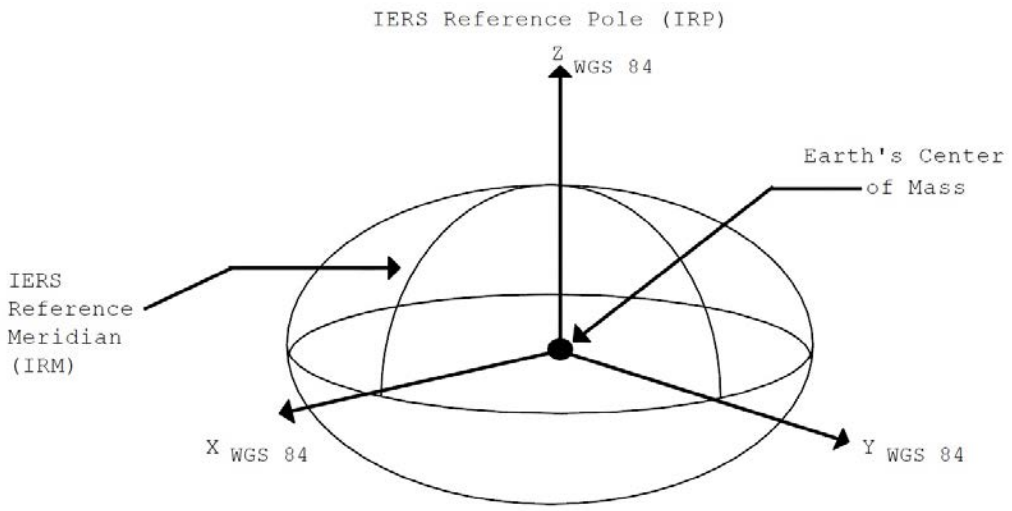


Figure 2.6: WGS-84 model [22]. The x and y axes rotate with the Earth while the z-axis remains stationary through the true North Pole.

a description of optimal estimation in inertial navigation systems, starting with its requisite mathematical system model.

2.5 System Modeling

Before optimal estimation can be accomplished, the system of interest must be represented mathematically. The first step in this process is to construct differential equations that represent the system dynamics. The second step is determining a mathematical representation of the various noise sources. The third step is determining what simplifications can be made while still maintaining a reasonable level of fidelity. Although navigation engineering is a hard science, decisions like this require judgement that may appear as much an art as a science. For the simplest construction and calculation, systems are assumed to be Linear Time-Invariant (LTI). If the equations must remain nonlinear, special methods, to be discussed later, are utilized for optimal estimation.

After a set of differential equations have been determined, they can be represented in state-space as

$$\dot{\mathbf{x}}(t) = \mathbf{F}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t) + \mathbf{G}(t)\mathbf{w}(t) \tag{2.18}$$

where $\mathbf{F}$(t) is the state dynamics matrix, $\mathbf{x}$(t) is the state vector, $\mathbf{B}$(t) is the input influence matrix, $\mathbf{u}(\mathbf{t})$ is the deterministic input vector, $\mathbf{G}$(t) is the noise-influence matrix, and $\mathbf{w}$(t) is the noise vector, usually characterized as White Gaussian Noise (WGN) [20]. The $\mathbf{F}$, $\mathbf{B}$, and $\mathbf{G}$ matrices are time-invariant in many cases, leading to a simpler solution. The $\mathbf{x}(t)$ vector contains the states of interest upon which the differential equations were written. It should be noted that Equation (2.18) is a linear equation. The system could be described more generally as

$$\dot{\mathbf{x}}(t) = f[\mathbf{x}(t), \mathbf{u}(t), t] + \mathbf{G}(t)\mathbf{w}(t) \tag{2.19}$$

where the function $f[\mathbf{x}(t), \mathbf{u}(t), t]$ may or may not be linear.

A unique solution to $\mathbf{x}(t)$ exists as

$$\mathbf{x}(t) = \mathbf{\Phi}(t, t_0)\mathbf{x}_0 + \int_{t_0}^{t} \mathbf{\Phi}(t, \tau)\, \mathbf{B}(\tau)\, \mathbf{u}(t) \mathrm{d}\tau \tag{2.20}$$

where $\mathbf{x}_0$ is the initial condition state vector and $\Phi(t, t_0)$ is the state transition matrix that propagates the homogenous portion of Equation (2.20) (not including the integral) from time t_0 to t [20].

In practical use on a computer, all mathematics are solved in the discrete domain. Equation (2.20) then becomes

$$\mathbf{x}_{k+1} = \mathbf{\Phi}_k \mathbf{x}_k + \mathbf{w}_k \tag{2.21}$$

where k is a discrete time, $k+1$ is next time interval, and $\mathbf{w}_k$ is the driven response at k due to the presence of the noise during the (t_k, t_{k+1}) interval [7].

In order to observe the state matrix at the next time interval, the discrete state transition matrix must be solved. This is accomplished using

$$\mathbf{\Phi}_k = e^{\mathbf{F}_k(dt)} \tag{2.22}$$

where $\mathbf{F}_k$ is the state dynamics matrix at time k and dt is the computer cycle time (sample time) [7].

2.5.1 Full State Representation. For navigation purposes, states are typically chosen to be *position* in three axes ($\mathbf{p}^n$), *velocity* in three axes ($\mathbf{v}^n$), *attitude* in three axes ($\mathbf{\Theta}^n$), *accelerometer biases* in three axes ($\mathbf{a}^b$), and *gyro biases* in three axes ($\mathbf{b}^b$) - for a total of 15 states. The accelerometer and gyro biases, always referenced in the b-frame for convenience, will be discussed in more detail in Section 2.6.2. A time invariant representation is

$$\mathbf{x} = \begin{bmatrix} \mathbf{p}^n \\ - - - \\ \mathbf{v}^n \\ - - - \\ \mathbf{\Theta}^n \\ - - - \\ \mathbf{a}^b \\ - - - \\ \mathbf{b}^b \end{bmatrix}_{15\times 1} \tag{2.23}$$

A partner noise vector for Equation (2.23) is

$$\mathbf{w} = \begin{bmatrix} \mathbf{w}_a^b \\ - - - \\ \mathbf{w}_b^b \\ - - - \\ \mathbf{w}_{a_{bias}}^b \\ - - - \\ \mathbf{w}_{b_{bias}}^b \end{bmatrix}_{12\times 1} \tag{2.24}$$

where $\mathbf{w}_a^b$ and $\mathbf{w}_b^b$ are additive accelerometer and gyro noises and $\mathbf{w}_{a_{bias}}^b$ and $\mathbf{w}_{b_{bias}}^b$ are accelerometer bias and gyro bias noises. These will described in further detail in Section 2.6.2.

2.5.2 Error State Representation. In practical use in navigation systems, an estimate of state *errors* is often considered instead of the full state. The high frequency dynamics experienced by an INS can be noisy and don't necessarily need to be modeled to accurately accomplish navigation estimation. The error state form of Equation (2.23) is

$$\delta\mathbf{x} = \begin{bmatrix} \delta\mathbf{p}^n \\ - - - \\ \delta\mathbf{v}^n \\ - - - \\ \psi^n \\ - - - \\ \delta\mathbf{a}^b \\ - - - \\ \delta\mathbf{b}^b \end{bmatrix}_{15\times 1} \tag{2.25}$$

where the attitude error vector (ψ) is defined as an array of small angle errors about the n-frame axes:

$$\psi^n = \begin{bmatrix} \psi_n \\ \psi_e \\ \psi_d \end{bmatrix}_{3\times 1} \tag{2.26}$$

The noise vector $\mathbf{w}$ remains the same for the error state representation. Derivation of the states' differential equations will be provided in Section 2.7.

2.6 Inertial Navigation Systems

The primary goal of inertial navigation is to determine position and velocity using an INS. Parameters that are a part of the mechanization, such as attitude, may also be desired for display.

The two basic components of an INS are a triad of accelerometers (consisting of a two-dimensional accelerometer along each axis) and a triad of gyros (fixed in each of the three planes corresponding to the axis convention). The accelerometers measure specific force in the b-frame ($\mathbf{f}^b$) which can be expressed as (Δv^b) after the acceleration of gravity has been removed and the computer cycle interval (dt) has been incorporated. Likewise, the gyros measure angular rates in the b-frame ($\Delta\theta^b_{ib}$) which can be expressed as ω^b_{ib} when dt is incorporated. Note that p, q, and r, the rotation rates about the body frame x, y, and z axes, are not the same as Euler angle rates.

The basic equation governing inertial navigation, known as the *navigation equation* is

$$\mathbf{a} = \mathbf{f} + \mathbf{g} \tag{2.27}$$

where $\mathbf{a}$ is the acceleration vector, $\mathbf{f}$ is the specific force vector, and $\mathbf{g}$ is the gravity vector. Once the acceleration vector is known, it is a simple matter to derive velocity and position:

$$\mathbf{v}(t) = \int \mathbf{a}(t)\, dt \tag{2.28}$$

$$\mathbf{p}(t) = \iint \mathbf{a}(t)\, dt \tag{2.29}$$

where dt is the variable of integration and is not related to the discrete sample time. Further discussions of position, velocity, and acceleration are assumed to be functions of time and will not include the postscript (t).

2.6.1 Strapdown INS Mechanization. The real-world case of inertial navigation in the n-frame is more complicated because it occurs in a noninertial, rotating reference frame. Figure 2.7 depicts the sequence of calculations that must be performed to calculate the acceleration vector for the n-frame. The navigation equation of Equation (2.27) becomes

$$\mathbf{a}_e^n = \mathbf{f}^n - \mathbf{g}^n \tag{2.30}$$

where

$$\mathbf{a}_e^n = \dot{\mathbf{v}}_e^n = \ddot{\mathbf{p}}_e^n \tag{2.31}$$

and $\mathbf{g}^n$ is the straight-down gravity vector.

The role of the attitude computer is to produce a new body-to-nav frame DCM for each iteration. This is done by solution of the equation

$$\dot{\mathbf{C}}_b^n = \mathbf{C}_b^n \boldsymbol{\Omega}_{nb}^b \tag{2.32}$$

which can also be expressed in the discrete time domain as

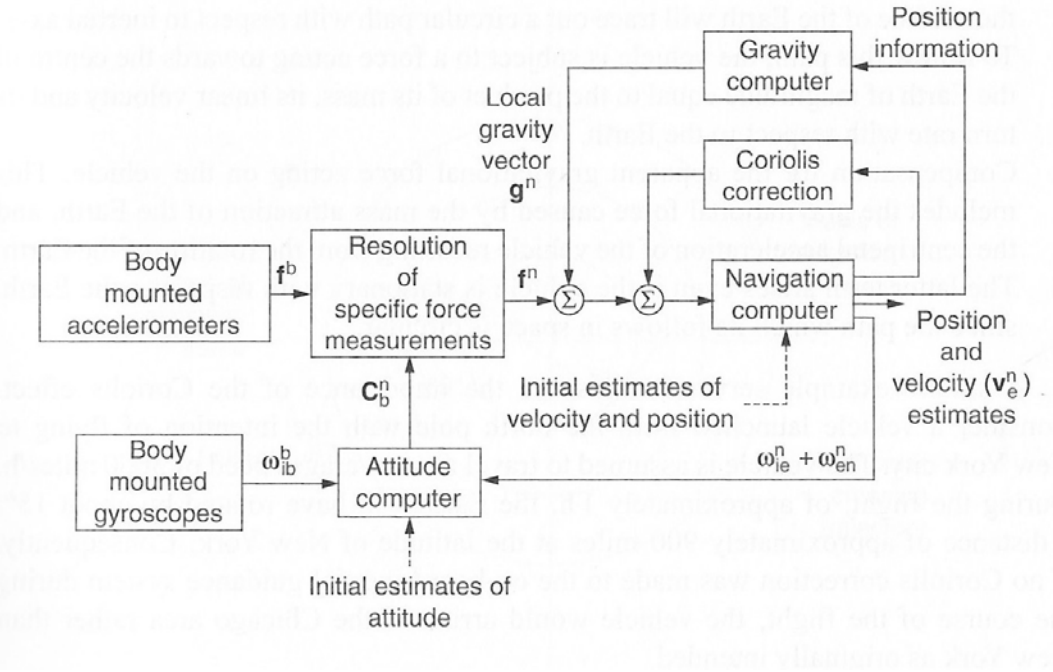


Figure 2.7: Strapdown INS mechanization in navigation frame [31]. The gyro measurements are processed to create a body-to-nav platform attitude to correctly interpret the accelerometer measurements in the n-frame.

$$\mathbf{C}_{b_{k+1}}^n = \mathbf{C}_{b_k}^n \left[\mathbf{I} + \mathbf{\Omega}_{nb}^b \, dt\right] \tag{2.33}$$

where $\mathbf{\Omega}_{nb}^b$ is the skew-symmetric form of ω_{nb}^b

$$\mathbf{\Omega}_{nb}^b = \omega_{nb}^b \times \tag{2.34}$$

and

$$\omega_{nb}^b = \omega_{ib}^b - \mathbf{C}_{n_k}^b \omega_{in}^n \tag{2.35}$$

where ω_{ib}^b is directly from the gyro and $\mathbf{C}_{n_k}^b$ is from the previous iteration (transposed). Additionally,

$$\omega_{in}^{n} = \omega_{ie}^{n} + \omega_{en}^{n} \tag{2.36}$$

where ω_{ie}^{n} is the sidereal rate (rotation rate of Earth) and ω_{en}^{n} is the transport rate (rate of the n-frame moving across the e-frame).

Now the $\mathbf{f}^{n}$ term of Equation (2.30) can be solved as

$$\mathbf{f}^{n} = \mathbf{C}_{b}^{n}\mathbf{f}^{b} \tag{2.37}$$

The $\mathbf{g}^{n}$ term of Equation (2.30) can be expressed as

$$\mathbf{g}^{n} = (2\omega_{ie}^{n} + \omega_{en}^{n}) \times \mathbf{v}_{e}^{n} + \mathbf{g}_{l}^{n} \tag{2.38}$$

where $(2\omega_{ie}^{n} + \omega_{en}^{n}) \times \mathbf{v}_{e}^{n}$ is a combination of Coriolis acceleration and centripetal force effects, and $\mathbf{g}_{l}^{n}$ is the local gravity vector. See [31] for an expanded explanation.

The last term of Equation (2.38) can be expressed as [23]

$$\mathbf{g}_{l}^{n} = \mathbf{g}^{n} - \frac{\Omega_{ie}^{n\,2}(R_o + h)}{2} \begin{bmatrix} sin(2L) \\ 0 \\ 1 + cos(2L) \end{bmatrix} \tag{2.39}$$

where $\mathbf{g}^{n}$ again refers to a straight-down vector and $\mathbf{g}_{l}^{n}$ refers to the local gravity vector, R_0 is the radius of the Earth, h is Above Ground Level (AGL) altitude, and Ω_{ie}^{n} is the scalar sidereal rate. The local gravity vector is not the same as the straight-down vector and incorporates the Earth's rotation and position on the Earth. All of the components required to determine the acceleration of the body in the n-frame with respect to the Earth have been solved according to Equation (2.30); velocity and position can now be determined by integration. Characterization of the accelerometer and gyro inherent errors will be examined next before construction of the governing differential equations.

2.6.2 Inertial Sensor Models. Both accelerometer and gyro measurements can be mathematically represented with additive noise and biases. The accelerometer equation is

$$\bar{\mathbf{f}}^b = \mathbf{f}^b + \mathbf{a}^b + \mathbf{w}_a^b \tag{2.40}$$

where $\bar{\mathbf{f}}^b$ is the measured specific force, $\mathbf{f}^b$ is the true specific force, $\mathbf{a}^b$ is the accelerometer bias, and $\mathbf{w}_a^b$ is the additive accelerometer noise. The gyro equation is

$$\bar{\omega}_{ib}^b = \omega_{ib}^b + \mathbf{b}^b + \mathbf{w}_b^b \tag{2.41}$$

where $\bar{\omega}_{ib}^b$ is the measured body angular rates, ω_{ib}^b is the true angular rates, $\mathbf{b}^b$ is the gyro bias, and $\mathbf{w}_b^b$ is the additive gyro noise.

The biases may be modeled as First Order Gauss-Markov (FOGM) processes as:

$$\dot{\mathbf{a}}^b = -\frac{1}{\tau_a}\mathbf{a}^b + \mathbf{w}_{a_{bias}}^b \tag{2.42}$$

where τ_a is the first-order accelerometer bias time constant and $\mathbf{w}_{a_{bias}}^b$ is the additive accelerometer bias noise, and

$$\dot{\mathbf{b}}^b = -\frac{1}{\tau_b}\mathbf{b}^b + \mathbf{w}_{b_{bias}}^b \tag{2.43}$$

where τ_b is the first-order gyro bias time constant and $\mathbf{w}_{b_{bias}}^b$ is the additive gyro bias noise.

2.7 Inertial Navigation Dynamics

The necessary mathematical framework has now been laid to write the error state differential equations to populate the dynamics matrix ($\mathbf{F}$).

2.7.1 Attitude Dynamics. The body-to-nav frame DCM can be approximated as [31]

$$\tilde{\mathbf{C}}_b^n \approx [\mathbf{I} - (\psi\times)]\mathbf{C}_b^n \tag{2.44}$$

Taking the time derivative of Equation (2.44) yields

$$\dot{\tilde{\mathbf{C}}}_b^n = -(\dot{\psi}\times)\mathbf{C}_b^n + [\mathbf{I} - (\psi\times)]\dot{\mathbf{C}}_b^n \tag{2.45}$$

Substituting Equation (2.32) into Equation (2.45) and solving for $(\dot{\psi}\times)$ yields

$$\dot{\psi}\times = [\mathbf{I} - (\psi\times)]\mathbf{C}_b^n\mathbf{\Omega}_{nb}^b\mathbf{C}_n^b - \tilde{\mathbf{C}}_b^n\tilde{\mathbf{\Omega}}_{nb}^b\mathbf{C}_n^b \tag{2.46}$$

Finally, substituting Equation (2.35) and Equation (2.41) into Equation (2.46) yields

$$\dot{\psi} = -[(\mathbf{C}_e^n\omega_{ie}^e)\times]\psi - \mathbf{C}_b^n\mathbf{b}^b - \mathbf{C}_b^n\mathbf{w}_b^b \tag{2.47}$$

2.7.2 Position and Velocity Dynamics. Starting from the most basic representation, position in the i-frame can be expressed as

$$\mathbf{p}^i = \mathbf{C}_e^i[\mathbf{p}_0^e + \mathbf{C}_n^e\mathbf{p}^n] \tag{2.48}$$

where $\mathbf{p}_0^e$ is the origin of the n-frame, expressed in the e-frame. Taking two time derivatives yields

$$\ddot{\mathbf{p}}^i = \mathbf{C}_e^i\mathbf{C}_n^e\ddot{\mathbf{p}}^n + 2\mathbf{C}_e^i\mathbf{\Omega^e}_{ie}\mathbf{C}_n^e\dot{\mathbf{p}}^n + \mathbf{C}_e^i(\mathbf{\Omega}_{ie}^e)^2[\mathbf{p}_0^e + \mathbf{C}_n^e\mathbf{p}^n] \tag{2.49}$$

Substituting the *navigation equation*, Equation (2.30), and Equation (2.31) into Equation (2.49) and solving for acceleration yields

$$\ddot{\mathbf{p}}^n = \mathbf{f}^n - 2\mathbf{C}_e^n \Omega_{ie}^e \mathbf{C}_n^e \dot{\mathbf{p}}^n - \mathbf{C}_e^n (\Omega_{ie}^e)^2 [\mathbf{p}_0^e + \mathbf{C}_n^e \mathbf{p}^n] + \mathbf{g}^n \tag{2.50}$$

and substituting the identity

$$\dot{\mathbf{p}^n} = \mathbf{v}^n \tag{2.51}$$

yields

$$\dot{\mathbf{v}}^n = \mathbf{C}_b^n \mathbf{f}^b - 2\mathbf{C}_e^n \Omega_{ie}^e \mathbf{C}_n^e \dot{\mathbf{p}}^n - \mathbf{C}_e^n (\Omega_{ie}^e)^2 [\mathbf{p}_0^e + \mathbf{C}_n^e \mathbf{p}^n] + \mathbf{g}^n \tag{2.52}$$

A calculated form, $\dot{\tilde{\mathbf{v}}}^n$, is constructed by adding accelerometer measurement and attitude errors and by substituting the gravity function [35], Equation (2.40), and Equation (2.47) to yield

$$\dot{\tilde{\mathbf{v}}}^n = \tilde{\mathbf{C}}_b^n \mathbf{f}^b - 2\mathbf{C}_e^n \Omega_{ie}^e \mathbf{C}_n^e \tilde{\mathbf{v}}^n + \mathbf{C}_e^n \mathbf{g}^e (\mathbf{p}_0^e + \mathbf{C}_n^e \tilde{\mathbf{p}}^n) \tag{2.53}$$

Substituting Equation (2.52) and Equation (2.53) into the acceleration error vector equation

$$\delta \dot{\mathbf{v}}^n = \dot{\tilde{\mathbf{v}}}^n - \dot{\mathbf{v}}^n \tag{2.54}$$

yields

$$\delta \dot{\mathbf{v}}^n = \mathbf{G}^n \delta \mathbf{p}^n - 2\mathbf{C}_e^n \Omega_{ie}^e \mathbf{C}_n^e \delta \mathbf{v}^n + (\mathbf{f}^n \times) \psi + \mathbf{C}_b^n \mathbf{a}^b + \mathbf{C}_b^n \mathbf{w}_a^b \tag{2.55}$$

where $\delta \mathbf{p}^n$ and $\delta \mathbf{v}^n$ are the error vectors denoting the difference between truth and calculated values.

All equations required for construction of a state-space dynamics model have now been derived.

2.8 Kalman Filtering

A Kalman filter is an optimal estimator that performs by means of an iterative algorithm. One of the most desirable features of the algorithm is that it not only maintains a mean estimate of the states of interest, but also produces and maintains a covariance matrix which corresponds to their uncertainties. During operation, the filter runs in two repeating steps: propagation and measurement update. Propagation occurs between every computer clock cycle. The expected value of the state is calculated along with the covariance matrix. When an update is available, the algorithm optimally weights how much to "believe" the new information. While simple in concept, designing a practical Kalman filter requires insight into proper system modeling to ensure the filter does not diverge or believe itself too well and not properly accept incoming measurements. It is also possible for the filter to become corrupted to where it believes all incoming measurements and has no memory of the past. The equations presented here will be in discrete form since that is most applicable to use in a computer program.

The system model, repeated from Equation (2.21) is

$$\mathbf{x}_{k+1} = \mathbf{\Phi}_k \mathbf{x}_k + \mathbf{w}_k \tag{2.56}$$

A Kalman filter in its basic form is a model-dependent filter and not adaptive; if the model does not fit the physical situation, it may yield poor results [18].

A model of measurements also exists:

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k \tag{2.57}$$

where $\mathbf{z}_k$ is the vector of measurements, $\mathbf{H}_k$ is the linear measurement observation matrix, $\mathbf{x}_k$ is the state vector, and $\mathbf{v}_k$ is the noise measurement vector.

Similar to Equation (2.19), Equation (2.57) can be generally expressed as

$$\mathbf{z}_k = h[\mathbf{x}_k] + \mathbf{v}_k \tag{2.58}$$

where $\mathbf{h}[\mathbf{x}_k]$ may or may not be linear.

2.8.1 Linear Kalman Filter. Although this research utilizes an unscented Kalman filter to deal with nonlinearities in propagation and measurement, the simpler case of a linear Kalman filter will be discussed to provide adequate background.

The discrete propagation equations carry the state and state uncertainty estimates from after the previous measurement (*a posteriori*) to right before the next measurement (*a priori*). If measurements are taken at a slower rate than the propagation rate, the propagation equations are simply repeated until a measurement is available. The state and state uncertainty propagation equations are:

$$\hat{\mathbf{x}}_k^- = \mathbf{\Phi}_{k-1}\hat{\mathbf{x}}_{k-1}^+ \tag{2.59}$$

$$\mathbf{P}_{xx,k}^- = \mathbf{\Phi}_{k-1}\mathbf{P}_{xx,k-1}^+\mathbf{\Phi}_{k-1}^{\mathbf{T}} + \mathbf{Q}_k \tag{2.60}$$

where the process noise matrix can be defined as

$$\mathbf{Q}_k = E\{\mathbf{w}_k\mathbf{w}_k^{\mathbf{T}}\} \tag{2.61}$$

where $E\{\bullet\}$ is the expectation operator.

The measurement update equations incorporate newly acquired information into the Kalman filter algorithm. The Kalman gain, $\mathbf{K}_k$, provides the optimum weighting of the new measurement. The update equations are:

$$\mathbf{K}_k = \mathbf{P}_{xx,k}^-\mathbf{H}_k^{\mathbf{T}}[\mathbf{H}_k\mathbf{P}_{xx,k}^-\mathbf{H}_k^{\mathbf{T}} + \mathbf{R}_k]^{-1} \tag{2.62}$$

where $\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^\mathbf{T} + \mathbf{R}_k$ is the *residual covariance*, with $\mathbf{R}_k$ defined as

$$\mathbf{R}_k = E\{\mathbf{v}_k\mathbf{v}_k^\mathbf{T}\} \tag{2.63}$$

where $\mathbf{v}_k$ is the uncertainty of the measurements and

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k[\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_k^-] \tag{2.64}$$

where $\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_k^-$ is the *residual*, and

$$\mathbf{P}_{xx,k}^+ = \mathbf{P}_{xx,k}^- - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_{xx,k}^- \tag{2.65}$$

The Kalman filter equations just described are for the linear case only, i.e. $f[\mathbf{x}_k, \mathbf{u}_k]$ is equivalent to $\mathbf{F}\mathbf{x}_k + \mathbf{B}_k\mathbf{u}_k$ and $h[\mathbf{x}_k]$ is equivalent to $\mathbf{H}_k\mathbf{x}_k$. If the system matrices are nonlinear, the equations can either be linearized beforehand which may result in highly inaccurate filter performance, or alternative filter types may be used. The most common filter type implemented to deal with nonlinearities is the Extended Kalman Filter (EKF) which linearizes the nonlinear system dynamic equations about nominal state points during each iteration. Another filter type that handles nonlinearities, and the method used in this research, the Unscented Kalman Filter (UKF) is described next.

2.8.2 Unscented Kalman Filter. The goal of the UKF is to improve upon the EKF by "capturing" higher-order effects of a Probability Density Function (PDF) An EKF truncates everything after the second term in the Taylor-series expansion when calculating the linear approximation. The UKF accomplishes state estimation by expressing the estimate as a collection of "carefully chosen" samples (particles) called *sigma points*. The main idea is that it is easier to approximate a probability distribution than it is to approximate an arbitrary nonlinear function or transformation [17]. The theory is similar to that of a Particle Filter (PF), except the particles

are not chosen at random. The UKF is also capable of capturing non-Gaussian statistics more accurately; however, all random variables will be assumed to be Gaussian in this research.

2.8.2.1 Unscented Transform. The original PDF is made useable to the UKF through the Unscented Transform (UT). A collection of $2L+1$ sigma points are defined as

$$\chi_0 = \hat{\mathbf{x}} \tag{2.66}$$

$$\chi_i = \hat{\mathbf{x}} + \sqrt{L+\lambda}\ (\sqrt[c]{\mathbf{P}_{xx}})_i\ \forall\ i \in [1, L] \tag{2.67}$$

$$\chi_i = \hat{\mathbf{x}} - \sqrt{L+\lambda}\ (\sqrt[c]{\mathbf{P}_{xx}})_i\ \forall\ i \in [L+1, 2L] \tag{2.68}$$

where $\hat{\mathbf{x}}$ is the mean state vector estimate, $\mathbf{P}_{xx}$ is the state covariance matrix, L is the number of states, and λ is a scaling parameter defined as

$$\lambda = \alpha^2(L+\kappa) - L \tag{2.69}$$

The $\sqrt[c]{\mathbf{P}_{xx}}$ refers to the Cholesky decomposition mathematical operation. Since a true square root of a matrix does not exist, the Cholesky decomposition is one of several methods to approximate the matrix square root. The considered matrix must be symmetric and positive-definite. The i subscript refers to the i^{th} column of the state covariance matrix.

The variable α is a user-specified parameter that changes the spread of the function and κ is a secondary tuning parameter, usually set to zero.

The sigma point weighting parameters are defined as

$$W_o^{(m)} = \frac{\lambda}{L+\lambda} \tag{2.70}$$

$$W_o^{(c)} = \frac{\lambda}{L+\lambda} + (1 - \alpha^2 + \beta) \tag{2.71}$$

$$W_i^{(m)} = W_i^{(c)} = \frac{1}{2(L+\lambda)} \tag{2.72}$$

where the (m) and (c) superscripts denote mean and covariance respectively, and β is a tuning parameter set equal to two in the Gaussian case.

The statistics of the sigma point populated PDF can be recovered using the equations

$$\hat{\mathbf{x}} = \sum_{i=0}^{2L} W_i^{(m)} \chi_i \tag{2.73}$$

$$\mathbf{P}_{xx} = \sum_{i=0}^{2L} W_i^{(c)} [\chi_i - \hat{\mathbf{x}}][\chi_i - \hat{\mathbf{x}}]^{\mathbf{T}} \tag{2.74}$$

Sigma points can be propagated through nonlinear dynamics to create transformed sigma points. These transformed sigma points can then make use of the given equations to calculate new statistics.

2.8.2.2 UKF Propagation. The concept of UKF propagation is that a collection of sigma points, consisting of a mean and statistically significant "surrounding points" are individually propagated through state dynamics equations, with the intent of being reassembled on the other side using mathematical tools that will create a new mean and covariance.

After the mean and covariance have been transformed to a collection of sigma points using the UT, the sigma points are passed though the state dynamics (mechanization equations for navigation applications):

$$\chi_{i,k+1} = f[\chi_{i,k}, \mathbf{u}_k, \mathbf{w}_k] \tag{2.75}$$

where they can be used to reconstruct the *a priori* mean and covariance:

$$\hat{\chi}^-_{i,k+1} = \mathbf{\Phi}(k+1,k)\hat{\chi}^+_{i,k} + \mathbf{B}_k\mathbf{u}_k \tag{2.76}$$

$$\mathbf{P}^-_{xx,k+1} = \mathbf{\Phi}(k+1,k)P^+_{xx,k}\mathbf{\Phi}^{\mathbf{T}}(k+1,k) + \mathbf{Q}_k \tag{2.77}$$

where $\hat{\chi}^-_{i,k+1}$ and $\mathbf{P}^-_{xx,k+1}$ are the sigma point state estimates and uncertainties after propagation in time, $\mathbf{\Phi}(k+1,k)$ is the state transition matrix, $\mathbf{B}_k$ is the input influence matrix, $\mathbf{u}_k$ is the input set, and $\mathbf{Q}_k$ is the discrete process noise.

2.8.2.3 UKF Measurement Update. The UT is also used to handle the nonlinear propagation as described in [12, 17].

Each sigma point is passed through the measurement equation:

$$\mathcal{Z}^-_{i,k} = h[\chi^-_{i,k}, \mathbf{v}_k] \tag{2.78}$$

to create the collection of predicted measurements, $\mathcal{Z}^-_{i,k}$. In a linear Kalman filter, this would be a one-dimensional vector of values; however, a measurement prediction is created for each sigma point in a UKF.

A mean predicted observation and measurement covariance are then calculated:

$$\hat{\mathbf{z}}_k = \sum_{i=0}^{2L} W_i^{(m)} \mathcal{Z}^-_{i,k} \tag{2.79}$$

$$\mathbf{P}_{zz,k} = \sum_{i=0}^{2L} W_i^{(c)} [\mathcal{Z}^-_{i,k} - \hat{\mathbf{z}}_k][\mathcal{Z}^-_{i,k} - \hat{\mathbf{z}}_k]^{\mathbf{T}} + \mathbf{R}_k \tag{2.80}$$

A cross-covariance matrix and Kalman gain are then computed as

$$\mathbf{P}_{xz,k} = \sum_{i=0}^{2L} W_i^{(c)} [\chi_{i,k} - \hat{\mathbf{x}}_k^-][\mathcal{Z}_{i,k}^- - \hat{\mathbf{z}}_k]^{\mathbf{T}} \tag{2.81}$$

$$\mathbf{K}_k = \mathbf{P}_{xz,k}\mathbf{P}_{zz,k}^{-1} \tag{2.82}$$

The *a posteriori* state estimate mean and covariance can be expressed as

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k(\bar{\mathbf{z}}_k - \hat{\mathbf{z}}_k) \tag{2.83}$$

$$\mathbf{P}_{xx,k}^+ = \mathbf{P}_{xx,k}^- - \mathbf{K}_k\mathbf{P}_{zz,k}\mathbf{K}_k^{\mathbf{T}} \tag{2.84}$$

where $\bar{\mathbf{z}}_k$ is the incoming measurement.

2.9 Digital Imaging

While the field of optics can be very complex and the mathematics very involved, the approach to understanding what is required for the test camera in this research will take every simplification available, to the level of fidelity required for this research. This section will discuss optical projection theory and lens distortion.

2.9.1 Optical Projection Theory. The use of a pinhole camera model is the fundamental assumption that greatly simplifies the concept of optics for this research. The term *focal length* refers to the distance from the center of the lens to the image plane when the size of the aperture (a hole through which light enters the camera) approaches zero. Figure 2.8 illustrates the projection of a an object in the distance, located at $\mathbf{s}^c$ that is projected onto the image plane at point $\mathbf{s}^{proj}$. The focal length can be used to define a ratio as

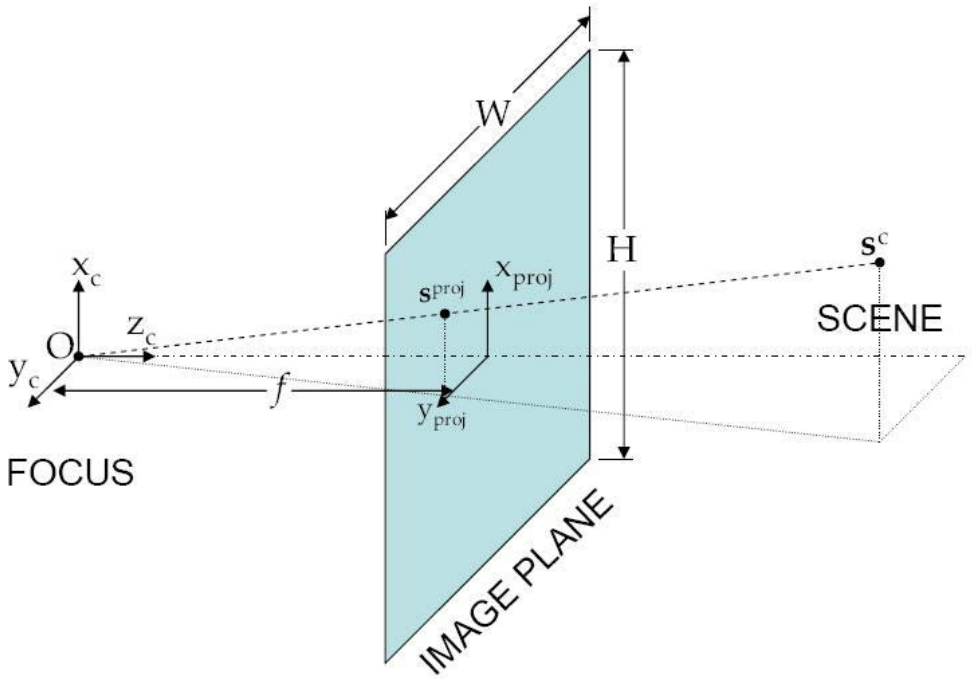


Figure 2.8: Three-dimensional image plane. H and W are physical measurements [35].

$$\mathbf{s}^{proj} = (\frac{f}{s_z^c})\mathbf{s}^c \tag{2.85}$$

where s_z^c is the scalar z-direction component of $\mathbf{s}^c$ in the c-frame. Note that W and H are physical measurements of the camera sensor itself. Figure 2.9 shows the same scenario looking directly onto the two-dimensional image plane.

Veth [35] shows that position vectors in the camera frame, denoted as $\mathbf{s}^c$ can be transformed to homogeneous pixel coordinates in the image plane frame by

$$\underline{\mathbf{s}}^{pix} = \frac{1}{s_z^c}\mathbf{T}_c^{pix}\mathbf{s}^c \tag{2.86}$$

with $\mathbf{T}_c^{pix}$ defined as

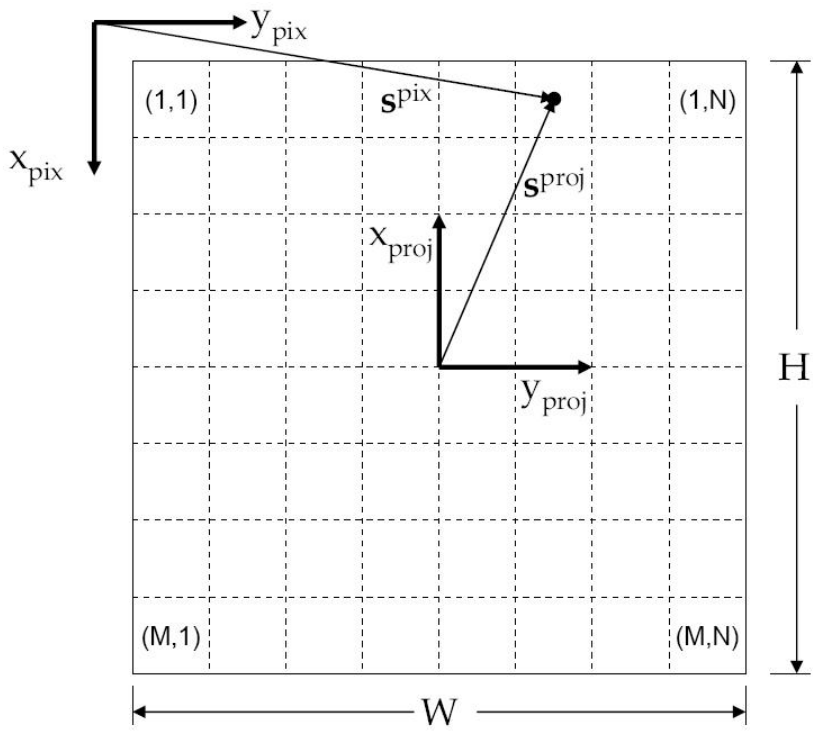


Figure 2.9: Image plane reference frame. H and W define the physical dimensions while M and N are the pixel dimensions [35]

$$\mathbf{T}_c^{pix} = \begin{bmatrix} -f\frac{M}{H} & 0 & \frac{M+1}{2} \\ 0 & f\frac{N}{W} & \frac{N+1}{2} \\ 0 & 0 & 1 \end{bmatrix} \tag{2.87}$$

where f is focal length, M and N are number of vertical and horizontal pixels on the image plane, and W and H are the physical measurements of the camera sensor. The inverse of $\mathbf{T}_c^{pix}$ may be used to convert from pixel coordinates to camera coordinates if s_z^c is known.

2.9.2 Lens Distortion. The inherent lens distortion of an actual camera is inconsistent with the pinhole approximation. Light is bent in complex ways through different areas of the lens before being focused on the focal plane. The pinhole camera

model simplifies the optics problems by assuming all light rays pass through a single point on the lens before creating a perfect, undistorted image on the focal plane.

It is, therefore, desired to remove gross distortion. While complex aberrations that may be present in a lens could more accurately be represented by high-order polynomials and complex three-dimensional vector fields, a suitable error model is the "Plumb Bob" distortion model [6] which separates the distortion into radial and tangential components.

An ideal nonhomogeneous two-dimensional representation (x,y) of a three-dimensional point (X,Y,Z) with no distortion can be shown as

$$\mathbf{x}_{no\ distortion} = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix} \tag{2.88}$$

To mathematically characterize radial distortion, the squared radius can be defined

$$r^2 = x^2 + y^2 \tag{2.89}$$

The components of distortion are then defined

$$\zeta_{radial} = 1 + kc_1 r^2 + kc_2 r^4 + kc_5 r^6 \tag{2.90}$$

$$\zeta_{tangential} = \begin{bmatrix} 2kc_3 xy + kc_4(r^2 + 2x^2) \\ kc_3(r^2 + 2y^2) + 2kc_4 xy \end{bmatrix} \tag{2.91}$$

where kc_1, kc_2, kc_3, kc_4, and kc_5 are distortion coefficients that must be solved, usually by calibration, in order to characterize the distortion.

A more complete representation of a point in an image with distortion applied can now be described as

$$\mathbf{x}_{distorted} = \zeta_{radial}\ \mathbf{x}_{no\ distortion} + \zeta_{tangential} \tag{2.92}$$

where $\mathbf{x}_{no\ distortion}$ is the original undistorted point from Equation(2.88). The practical goal for this research is then to create an *undistorted* image by applying these parameters in the form

$$\mathbf{x}_{no\ distortion} = \frac{\mathbf{x}_{distorted} - \zeta_{tangential}}{\zeta_{radial}} \tag{2.93}$$

2.10 Image Comparison Techniques

The problem of extracting information from a comparison of two or more images is one of both image processing and geometry involving transformations between two-dimensional image space and three-dimensional space where actual objects exist. This section discusses the image processing aspect and some contemporary techniques. The focus will be on two major categories of image comparison: pixel-based and feature-based. Other methods will be briefly mentioned to provide the reader with an overview of the state of the art.

2.10.1 Feature-based Methods. Feature-based techniques involve finding unique points of interest in an image and attempting to relocate these points in other images. Depending on how much the subsequent image may have changed in terms of translation, rotation, or affine movement, the feature-finding algorithm may not be able to find a matching feature point. Simple algorithms like the Harris-Stephens corner detector find matches where pixel-composition is very similar - such as another corner. More robust and computationally-costly algorithms like SIFT® are capable of finding matches on the surfaces of objects that have been decreased in size and partially changed in affine perspective. With an understood spatial relationship between feature points in a single image, and several confirmed feature-point matches between two different images, an understanding of the observed motion can be deducted.

Current vision and INS-coupled research solutions attempt to improve matching accuracy and speed by narrowing the match search area through statistical application of estimation and uncertainty. In other words, information is carried from one visual frame to the next that provides a best guess of where the feature point should lie, along with an uncertainty ellipse to constrain the search [35].

Only Harris-Stephens corner detection and SIFT® were used in the research, but many other methods exist. Some are relatively simple algorithms similar to the Harris-Stephens detector such as the Shi-Tomasi method [28] while others are more complex such as the Speeded Up Robust Features (SURF) algorithm [3].

2.10.1.1 Harris-Stephens Corner Detector. The Harris-Stephens corner detector is an improvement upon an earlier algorithm by Moravec. Conceptually, the method involves shifting a window and determining average intensity changes. Three cases are considered [16]:

1. Vertical and horizontal shifts of the window will result in small changes in average intensity if the image region encompassed by the window is approximately constant.
2. Shifts in the window along an edge will result in small intensity changes but shifts perpendicular to an edge will result in large change.
3. Vertical and horizontal window shifts in a region that contains a corner will have have large intensity changes in both directions.

The algorithm is designed to find corners but other means must be exploited to correlate found corners in one image with those in another image since there is no included specific feature descriptor.

2.10.1.2 SIFT®. The patented SIFT® algorithm, developed by David Lowe at the University of British Columbia is a robust, although computationally expensive, means of detecting points of interest in an image. Because of the

unique descriptors, they can be readily compared and matched with other images that contain the same feature points for many computer vision uses. A detailed mathematical explanation of the algorithm can be found in Lowe's publication [16], but the major steps of the algorithm will be briefly summarized.

1. *Scale-space extrema detection.* The image is convolved with Gaussian filters of different scales and scale-space extrema are found that persist through different scale changes. This is important because feature points generated by SIFT® are scale invariant, i.e. the same feature on an object should be detectable in image pairs that have different distances or zooming to the object.
2. *Keypoint localization.* The results of the previous step are pruned to remove unstable and low contrast candidates. The accuracy of remaining candidates are also improved by checking in a space around the candidates and searching for conflicting candidates.
3. *Orientation assignment.* A magnitude and direction are assigned to each feature point based on a calculated gradient of pixels in its local neighborhood.
4. *Keypoint descriptor.* A 128-dimension descriptor based on the previous steps is generated for each feature point. This ensures that each feature point is highly distinctive. This feature of SIFT® points makes it a very powerful tool in finding corresponding points in other images because of the high confidence that can be placed in the uniqueness of the keypoint descriptor to a particular feature point across different images.

For correlating points between images, matching is usually the slowest part of the process since point detection is fairly quick. There is no way to really determine if a feature point is correct in an image without searching for its twin in another image. It is also recommended that some logic outside of SIFT® be used to mitigate multiple matches. This is not in general a big problem but can be an issue when SIFT® is used within another specific-purpose algorithm that makes extensive use of geometry based on image pairing.

2.10.2 Pixel-based Methods. Pixel-based methods attempt to exploit more (ideally, all) of what can be seen. It is supposed that this is closer to how organic life perceives the environment and objects within it. While the potential for information extraction is greatly increased, the engineering implications also grow as every pixel in the image now has potential significance. Even so, some method of data reduction must be implemented to use the information. Some implemented techniques include Sum of Squared Differences (SSD), Sum of Absolute Differences (SAD), Normalized Cross Correlation (NCC), and variants of each. In general, pixel-based methods calculate disparity at each pixel within a neighborhood specified by an adjustable "window", which is a square neighborhood of pixels. A corresponding pixel in a second image is searched for by attempting to minimize error and maximize similarity.

Although these methods can be applied directly to images, depending on the specific software implementation, some type of pre-filtering or conditioning is often first applied. Examples include converting images from color to grayscale, thresholding (converting to a binary image), applying various gradient operators, and morphological techniques. Some, but not all of these techniques were used in this research. See [16] for further reading.

2.10.2.1 Sum of Absolute Differences. The SAD method subtracts pixels within a window between two images and then aggregates all of the differences within the window to provide a score for each pixel in the first image. SAD is the simples and least computationally burdensome method of pixel correlation. A tractable method for quick use of the results is to execute a two-dimensional summation on the scores to yield a single value. The lower the value, the higher the correlation of the two images. This concept is the basis for all of the pixel-based methods presented here.

The basic SAD equation is

$$SAD = \sum_{(i,j)\in W} |\mathbf{I}_1(i,j) - \mathbf{I}_2(x+i, y+j)| \tag{2.94}$$

where x and y are pixel coordinates, W is the chosen window, i and j are the pixels in the window being evaluated, and $\mathbf{I}_1$ and $\mathbf{I}_2$ are the first and second images. The algorithm is applied to every pixel in the images. Both images must be the same size.

A variant can be applied where the mean pixel intensity values within each window of each image are independently subtracted. This may produce better results when comparing images with wide contrast differences by normalizing them first. The zero mean SAD (ZSAD) equation is

$$ZSAD = \sum_{(i,j)\in W} |\mathbf{I}_1(i,j) - \bar{\mathbf{I}}_1(i,j) - \mathbf{I}_2(x+i,y+j) + \bar{\mathbf{I}}_2(x+i,y+j)| \tag{2.95}$$

where $\bar{\mathbf{I}}_1(i,j)$ and $\bar{\mathbf{I}}_2(i,j)$ are the mean intensity values of specified pixels.

Another method that attempts to normalize differences in the two compared images while calculating a correlation is the locally-scaled SAD (LSAD):

$$LSAD = \sum_{(i,j)\in W} |\mathbf{I}_1(i,j) - \frac{\bar{\mathbf{I}}_1(i,j)}{\bar{\mathbf{I}}_2(x+i,y+j)}\mathbf{I}_2(x+i,y+j)| \tag{2.96}$$

2.10.2.2 Sum of Squared Differences. The more popular SSD method squares and aggregates pixel differences within the window. The squared term adds a "penalty" for error distance and theoretically provides more accurate results. The general SSD formula is

$$SSD = \sum_{(i,j)\in W} (\mathbf{I}_1(i,j) - \mathbf{I}_2(x+i,y+j))^2 \tag{2.97}$$

Analogous to the zero mean SAD Equation (2.95), a zero mean SSD can be computed as

$$ZSSD = \sum_{(i,j)\in W} (\mathbf{I}_1(i,j) - \bar{\mathbf{I}}_1(i,j) - \mathbf{I}_2(x+i,y+j) + \bar{\mathbf{I}}_2(x+i,y+j))^2 \tag{2.98}$$

as well as a locally scaled SSD:

$$LSSD = \sum_{(i,j)\in W} \left(\mathbf{I}_1(i,j) - \frac{\bar{\mathbf{I}}_1(i,j)}{\bar{\mathbf{I}}_2(x+i,y+j)} \mathbf{I}_2(x+i,y+j) \right)^2 \tag{2.99}$$

2.10.2.3 Normalized Cross Correlation. The most complex method considered is NCC. It seeks to improve performance by also creating an increasing penalty, although more complex than that of SSD, based on error distance. The technique can be expressed as

$$NCC = \frac{\sum_{(i,j)\in W} \mathbf{I}_1(i,j)\mathbf{I}_2(x+i,y+j)}{\sqrt{\sum_{(i,j)\in W}(\mathbf{I}_1(i,j))^2 \sum_{(i,j)\in W}(\mathbf{I}_2(x+i,y+j))^2}} \tag{2.100}$$

2.10.3 Others. Several other methods of providing a means of correlation between images exist to include:

- Relate surfaces of objects in images using homography.
- Edge and line detection using gradient, Hessian operators, Sobel derivative operators.
- Phase correlation using a Fast Fourier Transform (FFT); match single peak in each image.
- Gradient cross-correlation.
- Template matching; a subimage of an object is rastered across a master image to find its location.
- Hough detection; can be tailored to find lines, circles, or other shapes.
- Wavelets; suited for edge detection.

It is also recommended that some means of removing outliers is used such as a median filter, Least Median of Squares Regression (LMedS) [26], or the Random Sample Consensus (RANSAC) algorithm [13]. These methods attempt to prevent obvious outlying data from influencing function and curve fitting to data.

2.11 Previous Work

Much work has been in accomplished in the field of computer vision and its application to the science of guidance, navigation, and control theory. This section summarizes a few efforts of particular interest, both academic and industrial.

2.11.1 Weaver. Weaver used a single KC-135 aircraft computer model to demonstrate the use of predictive rendering in the problem of autonomous aerial refueling [37]. An EKF was employed with inertial measurements to provide estimates to a MATLAB®-based synthetic view generation algorithm for comparison with available infrared imagery. He concluded that the most effective image processing technique was to use SSD for course correlation, followed by magnitude of gradient for fine tuning the position solution. An example of the gradient applied to the aircraft is shown in Figure 2.10. The use of the EKF drove the need for iterative perturbations around the predicted mean based on currently available state uncertainties, resulting in a slow process. As a result, Weaver recommended using a UKF for future work, creating a separate synthetic image based on each sigma point. This research uses that suggestion to create a statistically rigorous method of predictive rendering and comparison.

2.11.2 Ebcin. Ebcin developed a UKF-based algorithm for a tightly-coupled optical and inertial navigation system [12]. His algorithm is an error-state feedback system that uses full states for the strapdown mechanization portion of the algorithm. Once a collection of sigma points are computed about a nominal, each sigma point is transformed to and from whole-valued navigation state sigma points using simple addition and subtraction operations for position, velocity, accelerometer bias, and gyro bias. The whole-valued body-to-navigation frame DCMs are calculated by converting the angular errors to equivalent DCMs and the multiplying the DCM with the nominal body-to-navigation DCM. The results were successful, albeit with a heading bias that was not accounted for.

Figure 2.10: Gradient image of KC-135 from Weaver's research [37]. This method worked well for a lone object against a blank background (sky).

2.11.3 Veth. Veth's doctoral dissertation made several significant contributions toward the goal of "deep-level" image and inertial integration [35]. An online vision-aided inertial EKF-based algorithm was developed that not only estimated and corrected errors in the inertial sensor through help from the optical sensor, but streamlined the optical search space by implementing feedback from the filter. Previous techniques tended to be ad hoc and lacked statistical rigor. Although this constraining method is not used, this research borrows heavily from Veth's derivation of strapdown mechanization and digital imaging techniques. Veth concluded that the largest source of error came from EKF linearization, which this work attempts to avoid with use of the UKF.

2.11.4 Baumberg, Strecha, Tuytelaars, and Van Gool. The need for feature and/or object recognition in objects that demonstrate widely-changed aspects is obvious for applications of a computer vision in navigation. The authors of [2, 29, 32] investigate methods that allow for truly affine-invariant descriptors of local image

structures to be calculated and exploited for widely-spaced viewpoints. While their results show promise for keeping track of feature points when most of the same object surface is in view, it cannot deal with "turning a corner" and viewing surfaces from completely new perspectives. The research conducted for this thesis claims to partially solve this, although a virtual model must be first constructed. The work of these authors also lends itself to the emerging field of automatic three-dimensional object reconstruction. Combining future work of reconstruction and the work presented here could eventually lead to truly autonomous and robust precision navigation in any environment. For further reading on automatic three-dimensional reconstruction, see [1, 11, 21].

III. VANSPR Algorithm Development

This chapter details the novel approach of integrating synthetic visual data with actual visual data to aid the navigation solution of a UKF-based optimal estimation algorithm. The method is based on the SPR technique which consists of constructing synthetic views of a scene from the perspective of the test vehicle for comparison with actual images from an on-board camera. This predictively-rendered image is then compared to collected images using either feature-based or pixel-based comparison methods which serve to improve the accuracy of the correspondence search technique employed.

3.1 System Model

The basic dynamics of the system will be presented first, followed by a detailed walkthrough of the VANSPR algorithm.

3.1.1 Honeywell HG1700 INS. A well-performing optimal estimation algorithm is dependent on a correct model. The following INS parameters were used:

Table 3.1: Honeywell HG1700 tactical-grade IMU. These parameters were included in the model used by the Kalman filter [34]. PPM is Parts per Million.

Parameter	Units	HG1700
Sampling interval	ms	10.0
Gyro bias sigma	deg/hr	1.0
Gyro bias time constant	hr	2
Angular random walk	$deg/\sqrt{hr}$	0.3
Gyro scalefactor sigma	PPM	150
Accel bias sigma	m/s^2	0.0098
Accel bias time constant	hr	2
Velocity random walk	$m/s/\sqrt{hr}$	0.57
Accel scalefactor sigma	PPM	300

3.1.2 State Space Representation. The $\mathbf{F}(t)$, $\mathbf{G}(t)$, and $\mathbf{Q}(t)$ matrices, explained in the previous chapter, are constructed in the error state space form using

the error state form of the previously derived system differential equations. The differential equations are repeated here. The goal is to gather differential equations of the form of Equation (2.18), repeated here as

$$\delta\dot{\mathbf{x}}(t) = \mathbf{F}(t)\delta\mathbf{x}(t) + \mathbf{G}(t)\mathbf{w}(t) \tag{3.1}$$

where $\mathbf{B}(t)$ and $\mathbf{u}(t)$ are excluded as no external deterministic input is considered, and from Equation (2.25),

$$\delta\mathbf{x} = \begin{bmatrix} \delta\mathbf{p}^n \\ - - - \\ \delta\mathbf{v}^n \\ - - - \\ \psi^n \\ - - - \\ \delta\mathbf{a}^b \\ - - - \\ \delta\mathbf{b}^b \end{bmatrix}_{15\times 1} \tag{3.2}$$

and, from Equation (2.24)

$$\mathbf{w} = \begin{bmatrix} \mathbf{w}_a^b \\ - - - \\ \mathbf{w}_b^b \\ - - - \\ \mathbf{w}_{a_{bias}}^b \\ - - - \\ \mathbf{w}_{b_{bias}}^b \end{bmatrix}_{12\times 1} \tag{3.3}$$

The differential equations are as follows. Equation (2.51) is repeated and expressed in error state form as

$$\delta\dot{\mathbf{p}}^n = \delta\mathbf{v}^n \tag{3.4}$$

Equation (2.55) is repeated as

$$\delta\dot{\mathbf{v}}^n = \mathbf{G}^n\delta\mathbf{p}^n - 2\mathbf{C}_e^n\Omega_{ie}^e\mathbf{C}_n^e\delta\mathbf{v}^n + (\mathbf{f}^n\times)\psi^n + \mathbf{C}_b^n\mathbf{a}^b + \mathbf{C}_b^n\mathbf{w}_a^b \tag{3.5}$$

and Equation (2.47) is repeated as

$$\dot{\psi}^n = -[(\mathbf{C}_e^n\omega_{ie}^e)\times]\psi^n - \mathbf{C}_b^n\mathbf{b}^b - \mathbf{C}_b^n\mathbf{w}_b^b \tag{3.6}$$

along with the accelerometer and gyro biases from Equation (2.42) and Equation (2.43):

$$\delta\dot{\mathbf{a}}^b = -\frac{1}{\tau_a}\delta\mathbf{a}^b + \mathbf{w}_{a_{bias}}^b \tag{3.7}$$

$$\delta\dot{\mathbf{b}}^b = -\frac{1}{\tau_b}\delta\mathbf{b}^b + \mathbf{w}_{b_{bias}}^b \tag{3.8}$$

Now put in matrix form for n'-frame mechanization as

$$\mathbf{F}(t) = \begin{bmatrix} \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{G}^n & -2\Omega_{ie}^n & \mathbf{f}^n\times & \mathbf{C}_b^n & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & -\Omega_{ie}^n & \mathbf{0}_3 & -\mathbf{C}_b^n \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & -\mathbf{I}_3(\frac{1}{\tau_a}) & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & -\mathbf{I}_3(\frac{1}{\tau_b}) \end{bmatrix} \tag{3.9}$$

$$\mathbf{G}(t) = \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{C}_b^n & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & -\mathbf{C}_b^n & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \tag{3.10}$$

$$\mathbf{Q}(t) = \begin{bmatrix} \mathbf{I}_3(\sigma_{a_{rw}}) & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3(\sigma_{b_{rw}}) & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3\left(a_b\sqrt{\frac{2}{\sigma_a}}\right) & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3\left(a_b\sqrt{\frac{2}{\sigma_b}}\right) \end{bmatrix} \tag{3.11}$$

where $\sigma_{a_{rw}}$, $\sigma_{b_{rw}}$, σ_a, and σ_b are from Table 3.1.

The discrete form of $\mathbf{Q}(t)$ is calculated in Section 3.2.5.3

3.2 Algorithm Walkthrough

A complete walkthrough of the VANSPR algorithm will now be presented. The initial conditions and states are the best possible available: from TSPI. After the initial information handoff, similar to what a smart weapon such as a JDAM would receive from the carrier aircraft at launch, the algorithm functions independently and receives no further updates from any type of truth source.

3.2.1 Initial Body-to-Nav DCM. An initial body-to-nav DCM, $\mathbf{C}_b^n$, is required for propagation. It can be readily calculated based on the handoff of yaw(ψ), pitch(θ), and roll(ϕ). The $\mathbf{C}_b^n$ can be expressed as a series of matrix multiplications of Euler angle matrices by

$$\mathbf{C}_b^n = \mathbf{C}_1^n\mathbf{C}_2^1\mathbf{C}_b^2 \tag{3.12}$$

where

$$\mathbf{C}_1^n = \begin{bmatrix} cos\ \psi & -sin\ \psi & 0 \\ sin\ \psi & cos\ \psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.13}$$

$$\mathbf{C}_2^1 = \begin{bmatrix} cos\ \theta & 0 & sin\ \theta \\ 0 & 1 & 0 \\ -sin\ \theta & 0 & cos\ \theta \end{bmatrix} \tag{3.14}$$

$$\mathbf{C}_b^2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\ \phi & -sin\ \phi \\ 0 & sin\ \phi & cos\ \phi \end{bmatrix} \tag{3.15}$$

3.2.2 Sidereal Rotation in n-frame. It is also required to calculate the $\mathbf{C}_n^e$ matrix, which can reasonably be assumed constant over the 60-90 second interval of a locally-flown weapon that this research is considering. Starting with an origin defined by the starting point of a data collection run in WGS-84 format:

$$\mathbf{P}_0^{wgs} = [\phi_e\ \lambda\ alt_0]^{\mathbf{T}} \tag{3.16}$$

where ϕ_e is geodetic latitude, λ is geodetic longitude, alt_0 is absolute altitude. The conversion to $\mathbf{C}_{n0}^e$ is performed by

$$\mathbf{C}_e^n = \mathbf{C}_{n,lon}^{n,lat,lon}\mathbf{C}_e^{n,lon} = \begin{bmatrix} -sin\ \phi_e & 0 & cos\ \phi_e \\ -cos\ \phi_e\ sin\ \lambda & cos\ \lambda & -sin\ \phi_e\ sin\ \lambda \\ -cos\ \phi_e\ cos\ \lambda & -sin\ \lambda & -sin\ \phi_e\ cos\ \lambda \end{bmatrix} \tag{3.17}$$

and manipulated into the desired form by

$$\mathbf{C}_n^e = \mathbf{C}_e^{n\mathbf{T}} \tag{3.18}$$

It is then desired to have the rotation transformation in quaternion form. This conversion is well understood and can be found in [31]:

$$\mathbf{C}_n^e \;\Rightarrow\; \mathbf{q}_n^e \tag{3.19}$$

$$\mathbf{C}_e^n \;\Rightarrow\; \mathbf{q}_e^n \tag{3.20}$$

The sidereal rotation rate

$$\omega_{ie}^e = \begin{bmatrix} 0 & 0 & 7.292115 \times 10^{-5} \end{bmatrix}^{\mathbf{T}} rad/s \tag{3.21}$$

can now be transformed into the n-frame using the conversion of Equation (3.20) to yield

$$\omega_{ie}^n = \mathbf{q}_e^n \omega_{ie}^e \tag{3.22}$$

Finally, the skew symmetric form can be calculated for later use:

$$\Omega_{ie}^n = \omega_{ie}^n \times \tag{3.23}$$

3.2.3 UT Sigma Point Weights. As described previously in Chapter 2, Equations (2.70, 2.71, and 2.73) are repeated here as Equations (3.24, 3.25, and 3.26), with α set to one. This α setting was selected because it creates sigma points that have a sufficient distance from the mean sigma point for the purposes of offering varying viewpoints to the VANSPR algorithm. This will be discussed in more detail later. The weights are calculated by

$$W_o^{(m)} = \frac{\lambda}{L+\lambda} \tag{3.24}$$

$$W_o^{(c)} = \frac{\lambda}{L+\lambda} + (1 - \alpha^2 + \beta) \tag{3.25}$$

$$W_i^{(m)} = W_i^{(c)} = \frac{1}{2(L+\lambda)} \tag{3.26}$$

3.2.4 UKF Sigma Point Generation. Sigma points are re-generated on each iteration of the UKF cycle using Equations (2.66, 2.67, and 2.68) - repeated again here for convenience:

$$\chi_0 = \hat{\mathbf{x}} \tag{3.27}$$

$$\chi_i = \hat{\mathbf{x}} + \sqrt{L+\lambda}\ (\sqrt[c]{\mathbf{P}_{xx}})_i \ \forall\ i \in [1, L] \tag{3.28}$$

$$\chi_i = \hat{\mathbf{x}} - \sqrt{L+\lambda}\ (\sqrt[c]{\mathbf{P}_{xx}})_i \ \forall\ i \in [L+1, 2L] \tag{3.29}$$

where a scaling parameter λ is defined as

$$\lambda = \alpha^2(L + \kappa) - L \tag{3.30}$$

3.2.5 Strapdown Mechanization. An iterative cycle is then entered by the algorithm to propagate the navigation states in time and accept measurement updates. All functions within this cycle will be discussed. To the maximum extent possible, strapdown mechanization is performed using quaternions to reduce computational burden. (Ebcin showed using quaternions decreased processing time from in his UKF navigation algorithm from 410 to 198 seconds on a 60 second simulation [12]).

3.2.5.1 Attitude Propagation. Attitude propagation is initiated by transforming the body-to-nav sigma point quaternion rotations to body-to-earth sigma point quaternion rotations by

$$\chi_{\mathbf{q}_b^e} = \mathbf{q}_n^e \chi_{\mathbf{q}_b^n}^{\mathbf{T}} \tag{3.31}$$

where $\mathbf{q}_n^e$ is a 4×1 quaternion, while $\chi_{\mathbf{q}_b^n}$ is a $4\times L$ sigma point collection of quaternions. The sigma point collection is transposed

$$\chi_{\mathbf{q}_e^b} = \chi_{\mathbf{q}_b^{e\mathbf{T}}} \tag{3.32}$$

Numerically integrating the sidereal rate by

$$\theta_{ie}^e = \omega_{ie}^e \, dt = \begin{bmatrix} 0 & 0 & 7.292115 \times 10^{-5} \, dt \end{bmatrix}^{\mathbf{T}} radians \tag{3.33}$$

yields the Earth's rotation angle which can then be transformed into a collection of sigma points in the body frame using

$$\chi_{\theta_{ie}^b} = \chi_{\mathbf{q}_e^b} \theta_{ie}^e \tag{3.34}$$

Finally, the attitude representation is propagated by

$$\chi_{\theta_{nb_k}^b} = \Delta\theta_{n_{i,k}}^b - \chi_{\theta_{ie}^b} \tag{3.35}$$

where $\Delta\theta_{n_{i,k}}^b$ is the measurements from the gyros. A numerical derivative is then applied

$$\chi_{\omega_{nb_{k+1}}^b} = \frac{\chi_{\theta_{nb_k}^b}}{dt} \tag{3.36}$$

and an average body rotation rate is found by

$$\chi_{\omega_{nb\ average_{k+1}}^b} = \frac{\chi_{\omega_{nb_k}^b} + \chi_{nb_{k+1}}^b}{2} \tag{3.37}$$

The sigma point rotation angles from Equation (3.35) are converted to quaternion form

$$\chi_{\theta^b_{nb_k}} \Rightarrow \mathbf{q}^{b_k-}_{b_{k+1}} \tag{3.38}$$

rotated into the n-frame after the appropriate quaternion is calculated

$$\mathbf{q}^{n-}_{b_{k+1}} = \mathbf{q}^{n}_{b_k}\mathbf{q}^{b_k-}_{b_{k+1}} \tag{3.39}$$

to yield

$$\mathbf{q}^{n_k-}_{n_{k+1}} = \mathbf{q}^{n-}_{b_{k+1}}\chi^{+}_{\mathbf{q}^b_{n_k}} \tag{3.40}$$

3.2.5.2 Position and Velocity Propagation. A sigma point set of accelerations in the n-frame are constructed using accelerometer measurements by

$$\chi_{a^n_{k+1}} = \chi_{\mathbf{q}^n_b}\left(\frac{\Delta v^b_k}{dt}\right) + g^n - 2\Omega^n_{ie}\chi_{v^n_k} \tag{3.41}$$

In a trickle-down fashion, sigma point collections of velocity

$$\chi_{v^n_{k+1}} = \chi_{v^n_k} + \frac{1}{2}\left(\chi_{a^n_k} + \chi_{a^n_{k+1}}\right)dt \tag{3.42}$$

and position

$$\chi_{p^n_{k+1}} = \chi_{p^n_k} + \frac{1}{2}\left(\chi_{v^n_k} + \chi_{v^n_{k+1}}\right)dt \tag{3.43}$$

are also calculated.

3.2.5.3 Calculation of Φ_d and Q_d. From the $\mathbf{F}(t)$ matrix constructed in Equation (3.9), the discrete state transition matrix can be calculated using Equation (2.22), repeated here as

$$\mathbf{\Phi}_k = e^{\mathbf{F}_k(dt)} \tag{3.44}$$

The continuous time process noise is made available to the algorithm by converting it to discrete form by

$$\mathbf{Q}_k = \mathbf{G}(t)\mathbf{Q}(t)\mathbf{G}^{\mathbf{T}}(t) \tag{3.45}$$

using $\mathbf{G}(t)$ and $\mathbf{Q}(t)$ from Equations (3.10) and (3.11), respectively, and then to $\mathbf{Q}_{d_k}$ using a computational method from [20]:

$$\mathbf{Q}_{d_k} = \frac{1}{2}[\mathbf{\Phi}_k\mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^{\mathbf{T}} + \mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^{\mathbf{T}}\mathbf{\Phi}_k^{\mathbf{T}}]\, dt \tag{3.46}$$

3.2.5.4 Accelerometer and Gyro Bias Propagation. Before being passed through the INS mechanization equations, the accelerometer and gyros biases in sigma point format are removed:

$$\Delta v_{k+1}^b = \Delta v_k^b - \chi_{a_k^b} dt \tag{3.47}$$

$$\Delta \theta_{k+1}^b = \Delta \theta_k^b - \chi_{b_k^b} dt \tag{3.48}$$

The accelerometer and gyro biases are then propagated as

$$\chi_{a_{k+1}^b} = \mathbf{\Phi}_{a^b}(k+1,k)\chi_{a_k^b} \tag{3.49}$$

$$\chi_{b_{k+1}^b} = \mathbf{\Phi}_{b^b}(k+1,k)\chi_{b_k^b} \tag{3.50}$$

where $\mathbf{\Phi}_{a^b}$ and $\mathbf{\Phi}_{b^b}$ are the portions of the discrete state transition matrix relevant to the biases.

3.2.6 Calculating a priori Mean and Covariance. After propagation through the strapdown mechanization equations is complete, the sigma points are transformed back into a singular mean state vector and covariance matrix using the equations

$$\hat{\mathbf{x}}_k^- = \sum_{i=0}^{2L} W_i^{(m)} \chi_{i,k}^- \tag{3.51}$$

$$\mathbf{P}_{xx,k}^- = \sum_{i=0}^{2L} W_i^{(c)} [\chi_{i,k}^- - \hat{\mathbf{x}}_k^-][\chi_{i,k}^- - \hat{\mathbf{x}}_k^-]^{\mathbf{T}} + \mathbf{Q}_{d_k} \tag{3.52}$$

3.2.7 Current Position in WGS-84. After propagation of whole-valued states, it is desired to know where this takes the vehicle in terms of latitude, longitude, and altitude. That was defined previously to be in the unchanging n'-frame which has its origin where the data collection run began and the VANSPR algorithm began working.

The first step to determine WGS-84 coordinates is to convert the n'-frame x-y plane coordinates (Cartesian coordinates) into polar coordinates using

$$\theta_{Cartesian}^{polar} = \left(\frac{180}{2\pi}\right)\left(atan_2\left(\frac{p_y^{n'}}{p_x^{n'}}\right)\right) degrees \tag{3.53}$$

where $atan_2$ is the four quadrant arctangent, formulated to maintain sign integrity in all quadrants, and

$$r_{Cartesian}^{polar} = \sqrt{p_x^{n'2} + p_y^{n'2}}\ meters \tag{3.54}$$

The radius in meters is then converted into a distance in degrees. This calculation, specific to the Earth, is

$$r^{polar}_{Cartesian,deg} = \left(\frac{180}{2\pi}\right)\left(\frac{r^{polar}_{Cartesian}}{637100\ m}\right) degrees \tag{3.55}$$

Latitude can then be determined by

$$\phi_1 = asin\ [sin\ \phi_0 + cos\ r^{polar}_{Cartesian,deg} + cos\ \phi_0\ sin\ r^{polar}_{Cartesian,deg}\ cos\ \theta^{polar}_{Cartesian}]\ degrees \tag{3.56}$$

and longitude by

$$\lambda_1 = \lambda_0 + atan\left[\frac{sin\ r^{polar}_{Cartesian,deg}\ sin\ \theta^{polar}_{Cartesian}}{cos\ \phi_0\ cos\ r^{polar}_{Cartesian,deg} - sin\ \phi_0\ sin\ r^{polar}_{Cartesian,deg}\ cos\ \theta^{polar}_{Cartesian}}\right] degrees \tag{3.57}$$

3.2.8 Measurement Update. The key to the VANSPR algorithm is the measurement update. Updating a state, such as position or velocity, directly with a GPS position or Doppler velicometer is not difficult. However, when a state is being updated indirectly, such as through a vision-aiding algorithm, more labor is required in the construction of the measurement model.

3.2.8.1 Measurement Model. The UKF measurement model can be described by

$$\mathcal{Z}^{-}_{i,k} = \mathbf{h}[\chi^{-}_{i,k},\ \mathbf{v}_k] \tag{3.58}$$

where $\mathcal{Z}^{-}_{i,k}$ is the sigma point collection of measurements as they appear after the state sigma points have passed through the measurement system. Real-world measurements are not perfect, so statistically characterized measurement noise is added as $\mathbf{v}_k$.

In this research, the measurement model, h, simply passes the desired sigma point values through. The complete state sigma point collection can be described as the single matrix

$$\chi = \begin{bmatrix} \chi_{p^{n'}} \\ - - - \\ \chi_{v^{n'}} \\ - - - \\ \chi_{\Theta^{n'}} \\ - - - \\ \chi_{a^{b,n'}} \\ - - - \\ \chi_{b^{b,n'}} \end{bmatrix}_{15\times\ (2L+1)} \tag{3.59}$$

The measurement model selects the position sigma points, resulting in a $3 \times\ (2L+1)$ matrix

$$\mathcal{Z}_{i,k}^{-} = [\chi_{v^{n'}}]^{\mathbf{T}}_{3\times\ (2L+1)} \tag{3.60}$$

3.2.8.2 Sigma Point Image Comparison. With a few deviations that will be described in detail shortly, the main idea of the update phase of the algorithm is to compare synthetically-generated images based on relevant sigma points to actual camera images. In the 15-state UKF used here, it does not make sense for all state sigma point subsets to generate images. For example, changes in an image caused by accelerometer bias perturbations would be visually imperceptible in the short-term. It would also be difficult to truly establish observability of state impact. In other words, knowing *which* combination of state changes contributed to the new image is of the utmost importance, and it is crucial to avoid giving "credit" to the wrong states.

In consideration of this, the first simplification is to consider only the effects of position perturbations as commanded by the sigma point collection. The angle drift of the HG1700 IMU is negligible over the short periods of time (approximately 60 seconds) considered for the weapon trajectories. View changes due to the coupled effects of velocity, accelerometer bias, and gyro bias are not considered. Using only the three states of x, y, and z translation, seven sigma points are considered. The underlying behavior of the major position-influenced sigma points can be seen in Figure 3.1. Position states are realized as a nominal value sigma point (χ_0), and a $one-sigma$ point on each side. Lesser effects can be observed on sigma points beyond but are very small and were considered negligible.

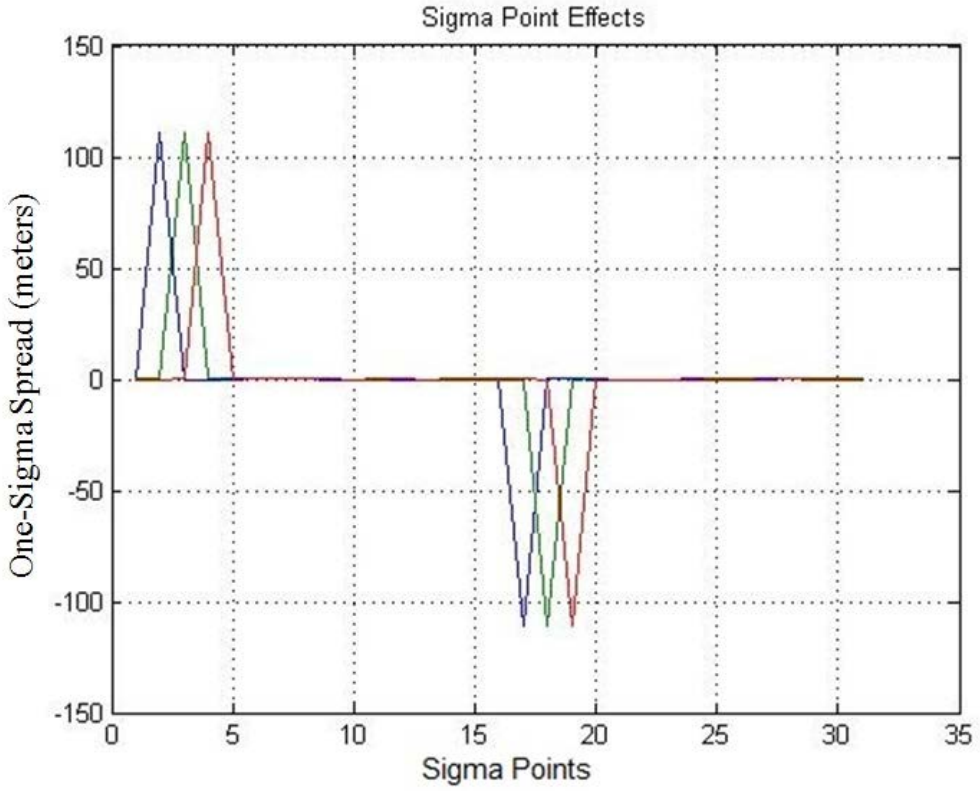


Figure 3.1: Sigma point spread. The primarily-influenced sigma points for the x position in the n'-frame are 1, 2, and 17. The y position is concerned with 1, 3, and 18 and the z position with 1, 4, and 19.

A further simplification is to assume that the altitude does not change drastically and that position changes can be satisfactorily considered in only one plane. With only two states, x and y, only five sigma points are required. The decision to make this reduction is based on empirical experience. It was too easy for the update mechanisms to continually take altitude updates and "run away" in altitude. Several workarounds were implemented to mitigate this, but the final and best solution was simply to not take altitude updates. This assumption may not be the best final answer, but provided a means to move on with the more critical aspects of the research. However, an issue with reducing the required number of observation points down to five is that they provide very little insight into a solution gradient, or direction. A workaround to this limitation was to add four *pseudosigma* points as shown in Figure 3.2. The location of the sigma points are set (by valuing the UKF parameter α to one) to $one - sigma$ of the current state covariance. This information is located in the current $\mathbf{P}_{xx,k}$ matrix. The ϵ_x, ϵ_y pairs are coordinates in the grid for referencing a specific (pseudo)sigma point. Northing and easting changes are made in accordance with the sigma points and a new synthetic image based on the perturbed position data is compared with an actual time-stamped image to extract positional error information.

3.2.8.3 SIFT® Matched Points as Information. As will be discussed in Section 5.3.1, the use of pixel-based methods (SAD, SSD, etc.) was discarded because of the need for continual manual adjustment from scenario to scenario. Similar problems existed in simple feature-point detection, line detection, and other methods. Hence, the use of SIFT® match points was determined to be the focus of information extraction for this research.

The actual method of detecting viable SIFT® points, as explained in Section 2.10.1.2, took approximately 5 seconds per image pair (800×800 resolution). The process of determining pairs amongst all the match from both images, however, was the most computationally burdensome (time-consuming) step of the measurement update. SIFT® matching was determined by computing the dot products between

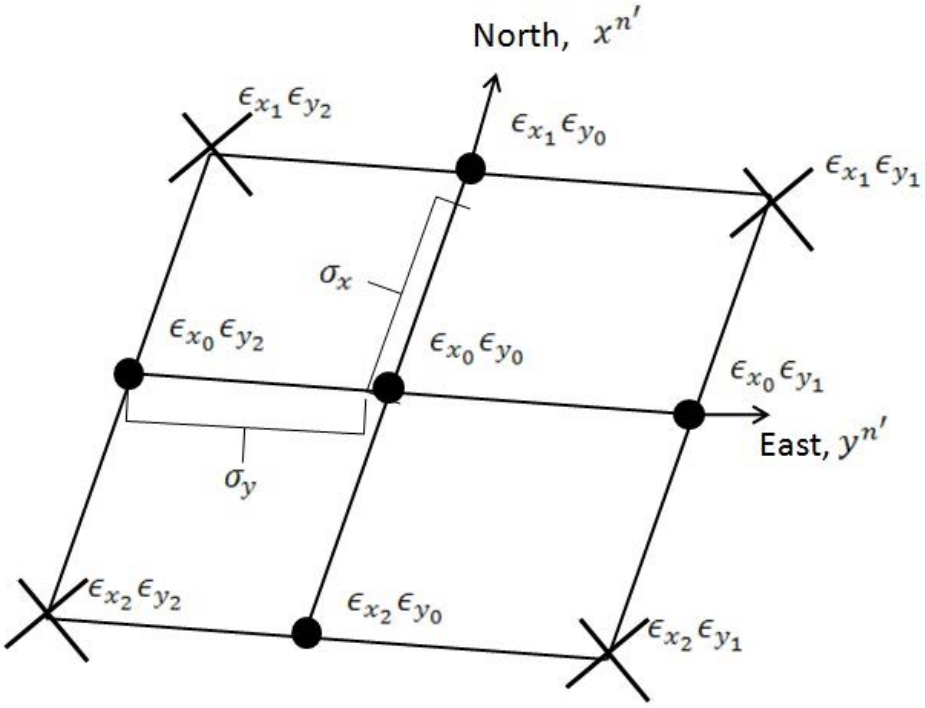


Figure 3.2: Augmented sigma point perturbation grid. The black circles represent sigma points and the black crosses represent *pseudosigma points*, added to provide greater visibility into error. The ϵ_x, ϵ_y pairs are coordinates in the grid. The grid does not consider altitude; an augmented sigma cube (not used successfully here) could be used to take altitude measurements as well.

all combinations of both images' descriptor vectors, sorting by angle size, and finding neighboring angles that were less than a specified threshold value. A SIFT® threshold value of 0.6 was generally determined to produce a viable number of SIFT® matches while minimizing false matches. In circumstances where "SIFT starvation", a lack of sufficient feature points to obtain viable results, is occurring, raising the threshold value may provide the required information while also raising the uncertainty of correct matches.

The collection of matched SIFT® pairs was intentionally reduced in the upper and lower vertical quarters of the image to avoid effects as described in Section 5.2.3 that skew the measurement by an artificial position "pulling". When SIFT® points were viewed in experimentation, situations were often observed where a large collection of SIFT® points were along the bottom fifth of the screen. An example image pair showing the reduced SIFT® set is shown in Figure 3.3.

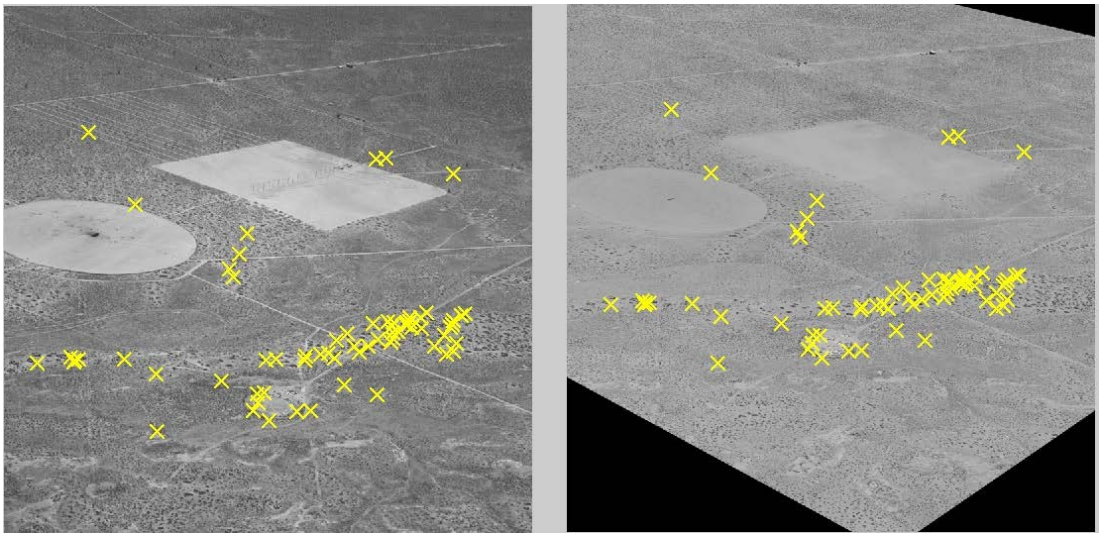


Figure 3.3: Reduced SIFT® collection; real (left), synthetic (right). SIFT® points in the upper and lower vertical quarters of the image are eliminated to prevent matching skew due to elevation error.

3.2.8.4 Taking the Measurement. The measurement step is a two-part process consisting of first determining which synthetic image most closely aligns with the actual camera image and then calculating a weighted shift direction vector to remove remaining observable pixel error. The goal of this second part is to align the SIFT® point matches as closely as possible and then determine the real-world position that would cause this image shift.

In the first part of the measurement step, the location coordinates of pixel match pairs in the two images differenced corporately. As shown in Figure 3.4, the pairs may not be in the exact same position relative to other match points in the same

image, but should maintain a relatively similar geometry overall if the algorithm has produced a useable synthetic view. Posed mathematically, the average x and y pixel shifts are calculated separately as

$$\Delta X_j^{pix} = \frac{1}{n}\sum_{i=1}^{n}(x_{s_i} - x_{r_i}) \tag{3.61}$$

$$\Delta Y_j^{pix} = \frac{1}{n}\sum_{i=1}^{n}(y_{s_i} - y_{r_i}) \tag{3.62}$$

and combined as

$$\gamma_j = \sqrt{(\Delta X_j^{pix})^2 + (\Delta Y_j^{pix})^2} \tag{3.63}$$

where γ_j is the pixel shift "score" of the j^{th} synthetic image and n is the number of matched SIFT® pairs. The lowest overall score across the synthetic images is declared the closest match:

$$\gamma_{match} = min\{\gamma\} \tag{3.64}$$

The value of this sigma grid location, as seen previously in Figure 3.2 is the error position state value

$$\delta\mathbf{p}^{n',\,match} = \epsilon_x, \epsilon_y(\gamma_{match}) \tag{3.65}$$

where $\delta\mathbf{p}^{n',\,match}$ is the closest perturbation based simply on the closest match. If the measurement calculation ended with this step, a measurement with the correct direction, but incorrect magnitude, would be obtained. Because two-dimensional pixel shift information is available from each of the nine (pseudo)sigma point comparisons, a refined measurement can be obtained without the requirement for further perturbation. It should be noted that the minimum x and minimum y shifts may actually

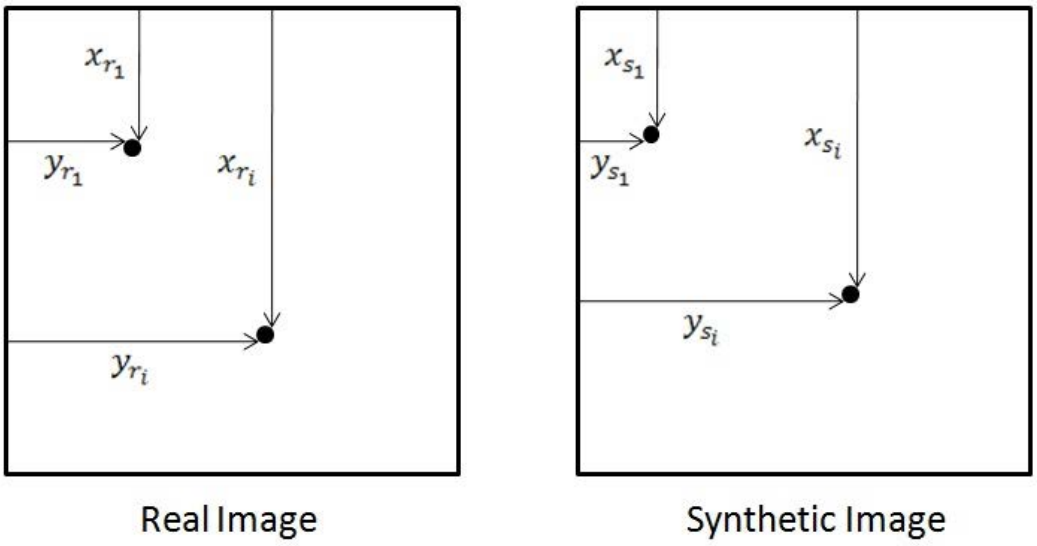


Figure 3.4: Pixel shift differencing. Coordinates of corresponding SIFT® points are subtracted and aggregated to characterize image error which can be interpreted to position error.

be from different (pseudo)sigma points, but the calculation of γ_{match} only concerns one (pseudo)sigma point. Therefore, a small amount of additional error exists, but determining it and removing it would require generating more synthetic views and repeating the whole process.

Now, the pixel error is removed in the dominant pixel error direction by a weighted magnitude. As an example, say that the dominant pixel error was in the x direction, so

$$\Delta X_{\gamma}^{pix} > \Delta Y_{\gamma}^{pix} \tag{3.66}$$

where $(\Delta X_{\gamma}^{pix}, \Delta Y_{\gamma}^{pix})$ are the x and y pixel shifts of γ_{match}.

Since the center and γ_{match} (pseudo)sigma point pixel shifts are known, a gradient can be established to interpolate or extrapolate where the zero pixel shift should occur. See Figure 3.5 for a graphical depiction of this example.

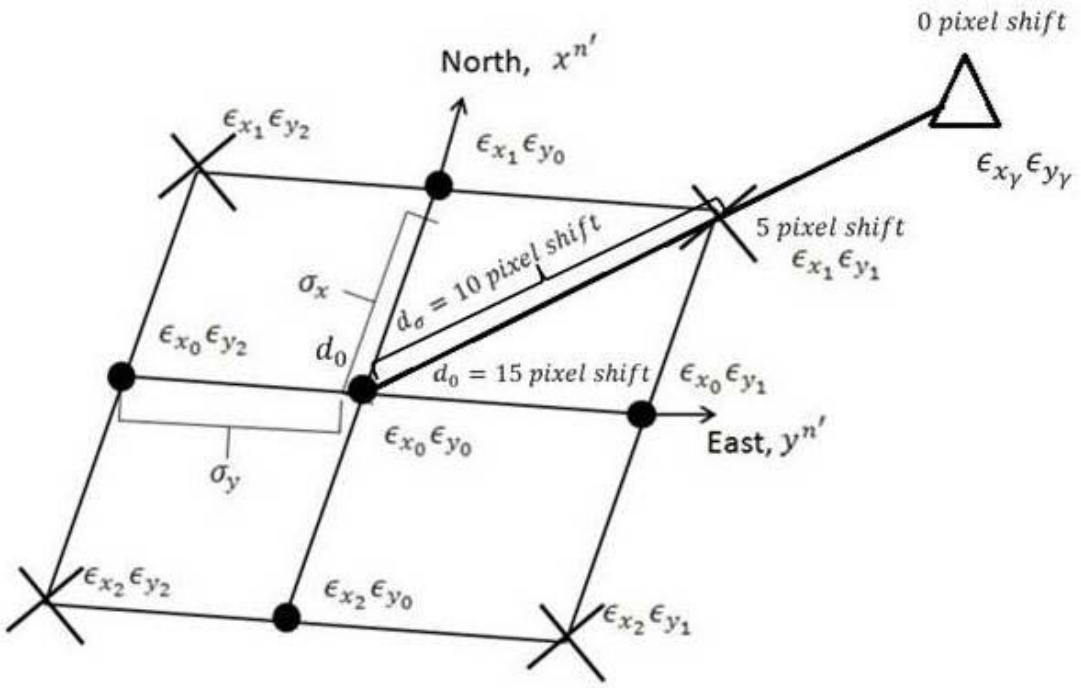


Figure 3.5: Example refinement of the update measurement with x being the dominant pixel error direction. The x pixel shift of the state estimate (center sigma point) is 15 pixels and the x pixel shift of the best match *pseudosigma* point is 5 pixels. Another 5 pixels in the same direction is required to achieve a zero-pixel shift. A ratio of $\left|\frac{d_0}{d_\sigma}\right|$ is calculated to determine fractional multiples of σ_x and σ_y to achieve the zero pixel shift point. Moving the position error by $1.5\sigma_x$ and $1.5\sigma_y$ should result in a zero pixel shift.

The zero distance is simply the pixel shift of the central (pseudo)sigma point:

$$d_0 = \Delta X_1^{pix}(\epsilon_{x_0}\epsilon_{y_0}) \tag{3.67}$$

where the first image is always the center (pseudo)sigma point corresponding to the mean estimate at $(\epsilon_{x_0}\epsilon_{y_0})$.

The sigma distance is the distance from the the center (pseudo)sigma point to the one determined previously as the best match:

$$d_\sigma = \Delta X_1^{pix}(\epsilon_{x_0}\epsilon_{y_0}) - \Delta X_\gamma^{pix}(\epsilon_{x_\gamma}\epsilon_{y_\gamma}) \tag{3.68}$$

The distance ratio is then defined as

$$\zeta_{0\gamma} = \left| \frac{d_0}{d_\sigma} \right| \tag{3.69}$$

The new error position estimate can now be calculated by scaling Equation 3.65 with Equation 3.69:

$$\delta\mathbf{p}^{n'} = \delta\mathbf{p}^{n',\,match}\zeta_{0\gamma} \tag{3.70}$$

and subtracting from the n'-frame position vector to create the optimum position measurement:

$$\bar{\mathbf{z}}_k = \mathbf{r}^{n'} - \delta\mathbf{p}^{n'} \tag{3.71}$$

3.2.8.5 Predicted Observation. After the selected positional sigma points are passed into the sigma point measurement collection via Equation (3.60), a mean predicted observation and measurement covariance can be calculated:

$$\hat{\mathbf{z}}_k = \sum_{i=0}^{2L} W_i^{(m)} \mathcal{Z}_{i,k}^- \tag{3.72}$$

$$\mathbf{P}_{zz,k} = \sum_{i=0}^{2L} W_i^{(c)} [\mathcal{Z}_{i,k}^- - \hat{\mathbf{z}}_k][\mathcal{Z}_{i,k}^- - \hat{\mathbf{z}}_k]^{\mathbf{T}} + \mathbf{R}_k \tag{3.73}$$

Calculation of a meaningful $\mathbf{R}_k$ matrix is non-trivial for an autonomous algorithm where the only allowed adaptations are internal and not manually adjusted. Several rigorous approaches were investigated utilizing pixel shift magnitudes, z-buffer distance for each pixel, and lens distortion characteristics. Weaver's method of $\mathbf{R}_k$

approximation was to fit an inverse Gaussian to his curve of errors built from perturbation fitting [37]. The method in this work, however, only generates five images in two Degrees of Freedom (DOF), plus the four cross-coupled *pseudosigma points*, for a total of nine. There is not enough information to build a meaningful Gaussian distribution curve. While one method would work well with one data set or against a certain target, a different approach would work better for another. Remembering the goal was to abstain from manually adjusting parameters, a long series of experimental observations led to the development of an empirical formula of weighted qualitative values. After further filter tuning, this too was abandoned in favor of a constant valued measurement noise which will be presented at the end of the next section. This may have the appearance of not being statistically rigorous; however, the desire to be mathematically pure must be balanced with the practical goal of engineering and tuning of an optimal estimator. The development of this empirical formula follows.

3.2.8.6 Measurement Noise Characterization. The presented $\mathbf{R}_k$ matrix consists of three calculated quantities for each direction: pixel spread, average pixel shift, and the number of useable SIFT® matches.

Pixel spread is calculated by first finding the minimum values in each of the three rows (columns for y-values) shown in Figure 3.2:

$$\upsilon_x = \begin{bmatrix} min\ \{\epsilon_{x_1}\} \\ min\ \{\epsilon_{x_0}\} \\ min\ \{\epsilon_{x_2}\} \end{bmatrix} \tag{3.74}$$

$$\upsilon_y = \begin{bmatrix} min\{\epsilon_{y_1}\} \\ min\ \{\epsilon_{y_0}\} \\ min\ \{\epsilon_{y_2}\} \end{bmatrix} \tag{3.75}$$

A pixel spread parameter is then calculated to find the widest difference between the three rows (or columns) by

$$\mu_x = max\ \{\upsilon_x\} - min\ \{\upsilon_x\} \tag{3.76}$$

$$\mu_y = max\ \{\upsilon_y\} - min\ \{\upsilon_y\} \tag{3.77}$$

Next, an overall average pixel shift in each direction is calculated by taking *another* average of the previously calculated (Equations (3.61) and (3.62)) image average pixel shifts:

$$\eta_x = \frac{1}{9}\sum_{j=1}^{9} \Delta X_j^{pix} \tag{3.78}$$

$$\eta_y = \frac{1}{9}\sum_{j=1}^{9} \Delta Y_j^{pix} \tag{3.79}$$

The third parameter, SIFT® weight score (W_{SIFT}), is invented according to Table 3.2. A minimum of 10 SIFT® point matches are required or the synthetic measurement in question is ignored. These weighted scores were determined from experimental observation; feature matches of about 150 or more provided the best results.

Table 3.2: SIFT® weight assignments. As implemented in the measurement noise equation, measurement uncertainty will decrease as the pool of SIFT® points increases, provided the SIFT® threshold is set to a level that does not allow for excessive mismatches.

SIFT® Weight Score	SIFT® Points Found (Windowed)
3	> 149
2	> 49
1	> 9
0 (no update)	< 9

Finally, the empirical measurement noise equations are presented as

$$\sigma_x = 0.15\left(\frac{100}{\mu_x}\right) + 0.7\left(\frac{\eta_x}{2}\right) + 0.15\left(\frac{20}{W_{SIFT}}\right) \tag{3.80}$$

$$\sigma_y = 0.15\left(\frac{100}{\mu_y}\right) + 0.7\left(\frac{\eta_y}{2}\right) + 0.15\left(\frac{20}{W_{SIFT}}\right) \tag{3.81}$$

where

$$\mathbf{R}_k = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix} \tag{3.82}$$

This method seemed to produce the best results for a limited set of data. However, as more data was introduced, filter tuning experimentation revealed optimum filter performance was actually achieved by using a constant measurment noise of

$$\mathbf{R}_k = \begin{bmatrix} 20^2 & 0 \\ 0 & 20^2 \end{bmatrix} \; meters^2 \tag{3.83}$$

3.2.8.7 Completing the Update. A cross-covariance matrix and Kalman gain are then computed as

$$\mathbf{P}_{xz,k} = \sum_{i=0}^{2L} W_i^{(c)} [\chi_{i,k} - \hat{\mathbf{x}}_k^-][\mathcal{Z}_{i,k}^- - \hat{\mathbf{z}}_k]^{\mathbf{T}} \tag{3.84}$$

$$\mathbf{K}_k = \mathbf{P}_{xz,k}\mathbf{P}_{xz,k}^{-1} \tag{3.85}$$

followed by the *a posteriori* state estimate mean and covariance, expressed as

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k(\bar{\mathbf{z}}_k - \hat{\mathbf{z}}_k) \tag{3.86}$$

where $\bar{\mathbf{z}}_k - \hat{\mathbf{z}}_k$ is the *residual*, or difference between the expected measurement value and the actual measurement value.

$$\mathbf{P}^{+}_{xx,k} = \mathbf{P}^{-}_{xx,k} - \mathbf{K}_k \mathbf{P}_{zz,k} \mathbf{K}^{\mathbf{T}}_k \tag{3.87}$$

This chapter provided a detailed walkthrough the VANSPR algorithm. The next chapter will bridge the theory into practical application in discussions of laboratory work, flight test planning and execution, and the creating of the computer target models.

IV. Laboratory Work, Flight Test, and Virtual World Model Construction

Prior to implementation of the VANSPR algorithm on flight test data, the ability to recreate virtual scenes from truth data position and attitude, followed by conception and execution of a flight test program to collect the necessary visual, inertial, and truth data was required. Additionally, large-scale VRML models were created for generation of the synthetic target views created by VANSPR. This chapter discusses the hardware, software, and other resources that were required, along with the methodology used in flight planning and model construction.

4.1 Laboratory Work

Laboratory work, conducted at AFIT, laid the foundation and built confidence in the fundamental techniques that would be used by the mature VANSPR algorithm after flight test at TPS. This work consisted of establishing a small-scale truth position source, creating a computer model of a simulated target, and generating synthetic viewpoints based on truth position and attitude that accurately match real photographs taken by a test camera.

4.1.1 Vicon® System. Position and attitude truth data was obtained by a Vicon® precision tracking system. A Vicon® camera, shown in Figure 4.1(a) tracks small balls made of reflective tape. When used in a constellation of 10 cameras, as partially shown in Figure 4.1(c), position of the reflective balls to an accuracy of 1 millimeter was achievable. Figure 4.1(b) shows the test camera, a 1280×960 resolution Prosilica GC1290C (color) scientific camera and a board that was glued on the top with tracking balls. The Vicon® system did not directly provide attitude information, so the attached board contained several tracking balls that, when individually and accurately triangulated, could produce a calculated three-axis attitude. The origin translation between the center of the camera's optical plane and the ball immediately

(a) Front view of one Vicon® camera.

(b) Board with four tracking spheres glued to test camera.

(c) Three of the 10 cameras arranged in a circle around the test area in the Vicon® lab.

Figure 4.1: Vicon® system at AFIT. The system of 10 cameras provided precision tracking of specially marked objects to a precision of 1 millimeter.

above it were measured to a 1 mm accuracy and fine rotational biases were not determined.

The Vicon® system proved to be a convenient method of gathering precise position and attitude data for validation of the algorithms' ability to recreate the scene synthetically. Small position and angular errors that were noted during software viewpoint recreation testing were difficult to elminate. This was partially attributed to the "tyranny of small scale," referring to the effect of very small errors resulting in large visual changes because of the small-scale differences between camera, target objects, and the distances between them.

4.1.2 Small-Scale Model Creation. The first step was creation of a simple target model, comprised of a small rug to represent the target environment and a

(a) Box "target" with visually distinctive markings.

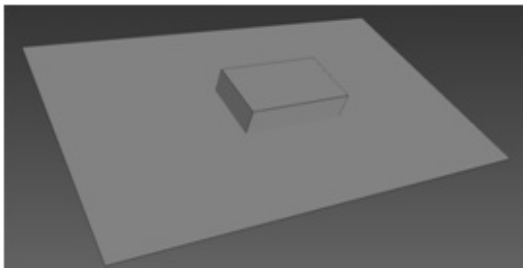
(b) Rug "target environment" with distinctive pattern markings.

(c) Simple combined prototype target in target environment.

Figure 4.2: Early target environment at AFIT's Vicon® lab. Individual objects were modeled in Google Sketchup Pro® and combined virtually.

box to serve as a three-dimensional target object within the target environment. Measurement and photographs of all surfaces of the objects were taken and the effects of lens distortion were removed; this process was especially germane to images taken during flight test and will be discussed later. The objects to be modeled are shown in Figure 4.2. Creation of the virtual environment was accomplished in Google Sketchup Pro®, which will be discussed in further detail in Section 4.2.8. The modeled shape of the target environment and the complete model with texture-mapped photographs applied are shown in Figure 4.3

4.1.3 Virtual Viewpoint Generation. The model generated in Google Sketchup Pro® was then exported as a VRML object. MATLAB®, which is capable of reading



(a) Target environment with no texture-mapping.

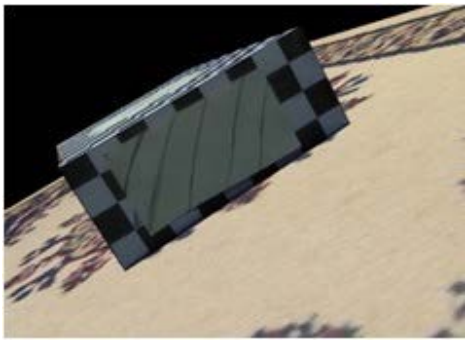
(b) Target environment after texture maps (photos of all object surfaces) have been added.

Figure 4.3: Construction of prototype target environment with target in Google Sketchup Pro®. The object shapes are created first, followed by texture-mapping photos to object surfaces.

this object format, was then used to manipulate the object in position and attitude. The common aerospace viewpoint conventions of Cartesian position and roll, pitch, and yaw differ from the MATLAB® implementation so special code had to be written to translate these parameters into those used by MATLAB®. Once this was written as a callable function, there were no further interpretation issues. Figure 4.4 shows three sample viewpoints generated in MATLAB®.

The VRML scripting language provides a near limitless means to add and modify three-dimensional computer models. One application is the addition of non-real objects for the purpose of visualization, such as the rocket shown in Figure 4.5 which was associated to the test camera. In experimental runs, a separate MATLAB® window showed a rocket trajectory towards the box with the physical test camera represented as the rocket. A practical future application for the creation of novel objects is the addition of new structures to existing target models based on updated intelligence. Unfortunately, MATLAB® only possesses very high-level access to VRML objects; therefore, changes such as this have to be written directly in VRML and then copied into the VRML world file. Special functions were written to take care of all this seamlessly in MATLAB®.

(a) Synthetic upside-down view of the prototype target environment.



(b) Synthetic ground-level view of the target object (box).

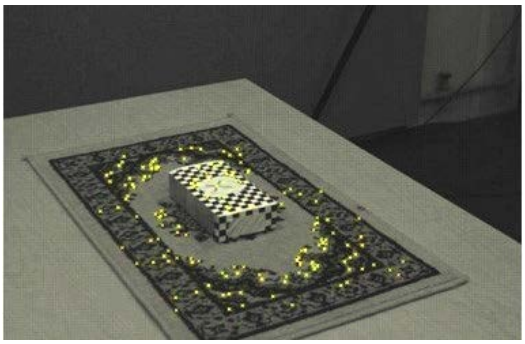
(c) Close-up synthetic view of box target.

Figure 4.4: Virtual views of the prototype target environment as viewed in MATLAB® using the 3D Simulink Animation Toolbox. None of these views were actual photographs but virtually created after the constituent objects were modeled.

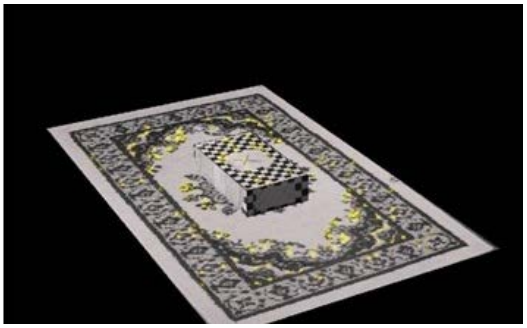
Figure 4.5: Synthetic view of prototype target environment with the addition of a fabricated rocket object that was created for inclusion in the virtual space. Novel scenes can be created combining real and imaginary objects; this technique could be used to add new structures to existing target models based on updated intelligence.

4.1.4 Pixel-based Methods Between Real and Synthetic Images. The pixel-based methods of SAD, ZSAD, LSAD, SSD, ZSSD, LSSD, and NCC were each applied to ten sample data sets of real images with five synthetically-generated perturbation images for each. The methods were used after Canny line detection was applied to the images. The perturbations were significantly different from each other. While the different methods produced a varied gradient of scores, all methods achieved 100% correct matching. This created the expectation that results would be similar for the test flight data.

4.1.5 SIFT® Matching Between Real and Synthetic Images. A picture of the VRML viewpoint, referred to as a *synthetic image*, can look very similar to the actual photograph taken from the same vantage point. Early experimentation with pixel-based methods, such as SSD, were highly successful in differentiating between which synthetic images provided the closest match with actual camera image. Additionally, feature-based methods such as SIFT® yielded excellent results. The SIFT® algorithm neither knew nor cared that both images were not actual photographs. Fig-



(a) Real image with SIFT® matches.



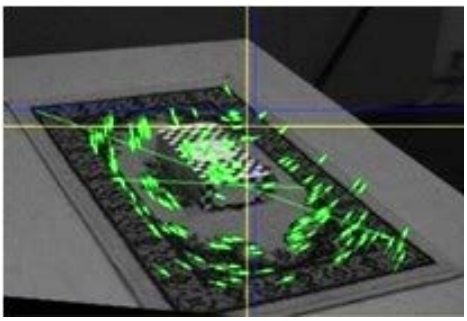
(b) Synthetically-rendered image with SIFT® matches.

Figure 4.6: SIFT® feature point matching between real and synthetic images. SIFT® neither knows nor cares that both images are not real photographs.

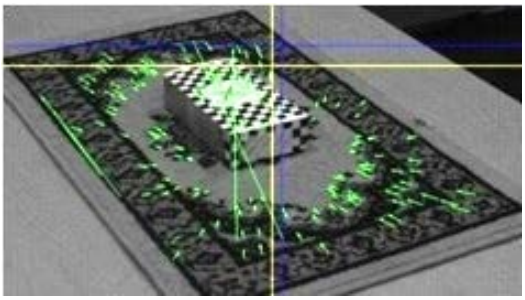
ure 4.6 shows a sample real and synthetic image pair with about 50 matching SIFT® pairs.

Early work also considered methods of exploiting the SIFT® pair correspondences through their relative geometry. Methods of determining translation, rotation, scaling, and combinations of these were developed and successfully demonstrated by intentionally injecting errors into the synthetic renderings and then allowing written error detection algorithms to find the deviations. The results were successful recognition of the error and quantification within 5%. An example of the method is depicted in Figure 4.7.

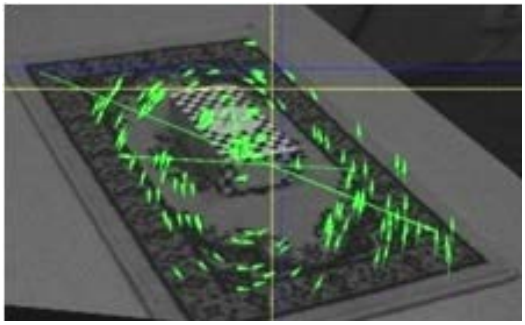
4.1.6 Fidelity of Synthetic Views. The most important question for use of virtual viewpoints in an algorithm that relies on them for correspondence with real images is to what extent do the synthetic images match the real thing when given perfect position and attitude information. A series of sample trajectories were created with the Vicon® system by "hand flying" the camera toward the rug-box target while ensuring the reflective tracking balls were visible to the Vicon® camera constellation. The resulting synthetic images were remarkably similar to the real images, with only minor differences noted at extremely close ranges. Figure 4.8 shows a transparent overlay of the synthetic image on the real image for several frames of one



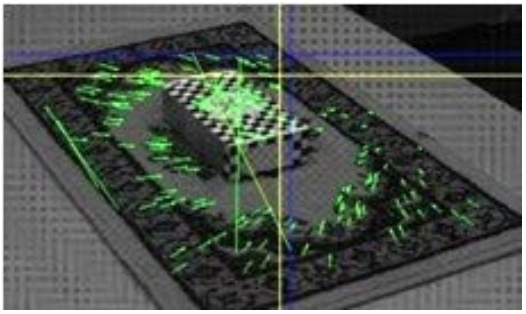
(a) Translation and rotation.



(b) Scaling.



(c) Rotation.



(d) Translation and scaling.

Figure 4.7: Detection of image change between real and synthetic views using SIFT® point vector flows. Changes were intentionally injected to observe effects and test error observability.

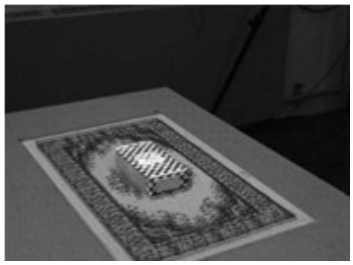
of the experimental trajectory runs. Very good matches show almost no perceptible difference while errors are revealed by a blurring effect where pixels don't line up correctly. Laboratory experiments were extremely useful in providing familiarization with the concepts, capabilities, and limitations of predictive rendering. The next step was a larger scale experiment: flight test.

4.2 Flight Test Hardware, Software, and Other Resources

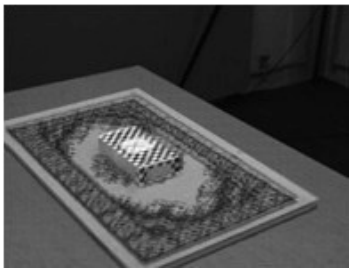
The successful collection of the necessary flight test data and creation of computer models required a well-planned hardware and software design. The basic lessons from previous camera-on-aircraft projects were implemented; however, most of the hardware and software solution was original and therefore designed from the ground up. Figure 4.9 shows a hardware schematic of the aircraft modification used for in-flight data collection.

4.2.1 C-12C "Huron" Aircraft. One C-12C "Huron" aircraft as shown in Figure 4.10, tail number 76-0161, was modified and flown to collect the required flight test data. Behind the cockpit, which included the pilot and copilot stations, a total of three passenger seats were installed for the flight test crew. An equipment rack was installed on the left side of the cabin which held a tower desktop computer, a laptop computer, a Time Code Generator (TCG), a truth source and INS unit, an ethernet network switch, and associated power supplies and cabling. A GPS antenna was installed on the outside of the aircraft. The Prosilica 4900 test camera was fixed to a custom mounting bracket which was mounted in a driftmeter port under the copilot's seat. One of the pilots' Multi-Function Displays (MFD) was modified to repeat the images being recorded by the test camera.

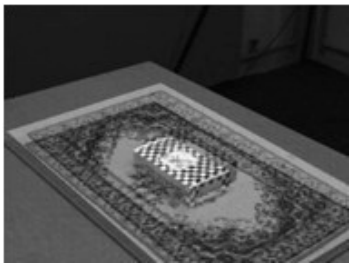
4.2.2 Prosilica 4900 Camera System. The Prosilica 4900 camera, serial number 02-2095A-06062, was a high-resolution (16 megapixel (MP), 4,872 horizontal x 3,248 vertical pixels) monochrome camera with a Kodak KAI-16000 sensor. The active image size of the sensor is 36.1 millimeters horizontal and 24.0 millimeters



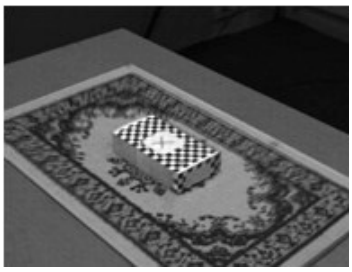
(a) Trajectory frame #1. Small errors are seen as blurring between the overlays.



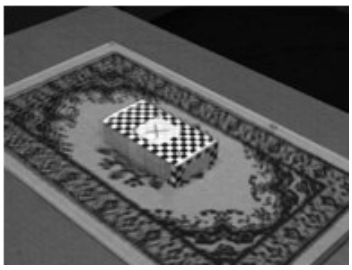
(b) Trajectory frame #2. Small errors are again seen as blurring between the overlays.



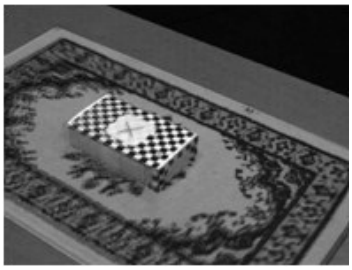
(c) Trajectory frame #3. More significant errors are detectable as differences between the images.



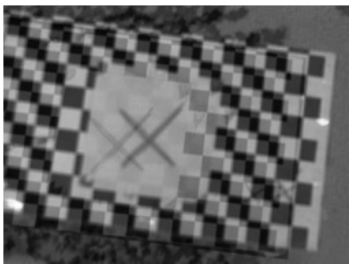
(d) Trajectory frame #4. Errors are very small and show an almost perfect match.



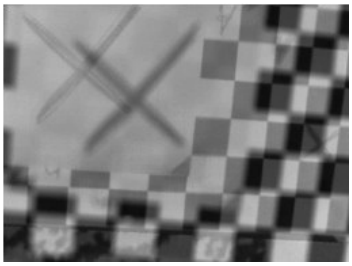
(e) Trajectory frame #5. Slight blurring in upper left corner shows discrepancy between real and virtual images



(f) Trajectory frame #6. Another good image pair with strong correlation.



(g) Trajectory frame #7. Closer view shows error more clearly.



(h) Trajectory frame #8. Error persists, which may be an error bias.

Figure 4.8: Selected frames from simulated weapon trajectory against the rug-on-box target environment showing transparent synthetic views overlaid on photographs. Vicon® position and derived attitude measurements were provided to the synthetic viewpoint rendering routine to create the synthetic views. Overall, synthetic viewpoint images strongly correlated with the actual photographs.

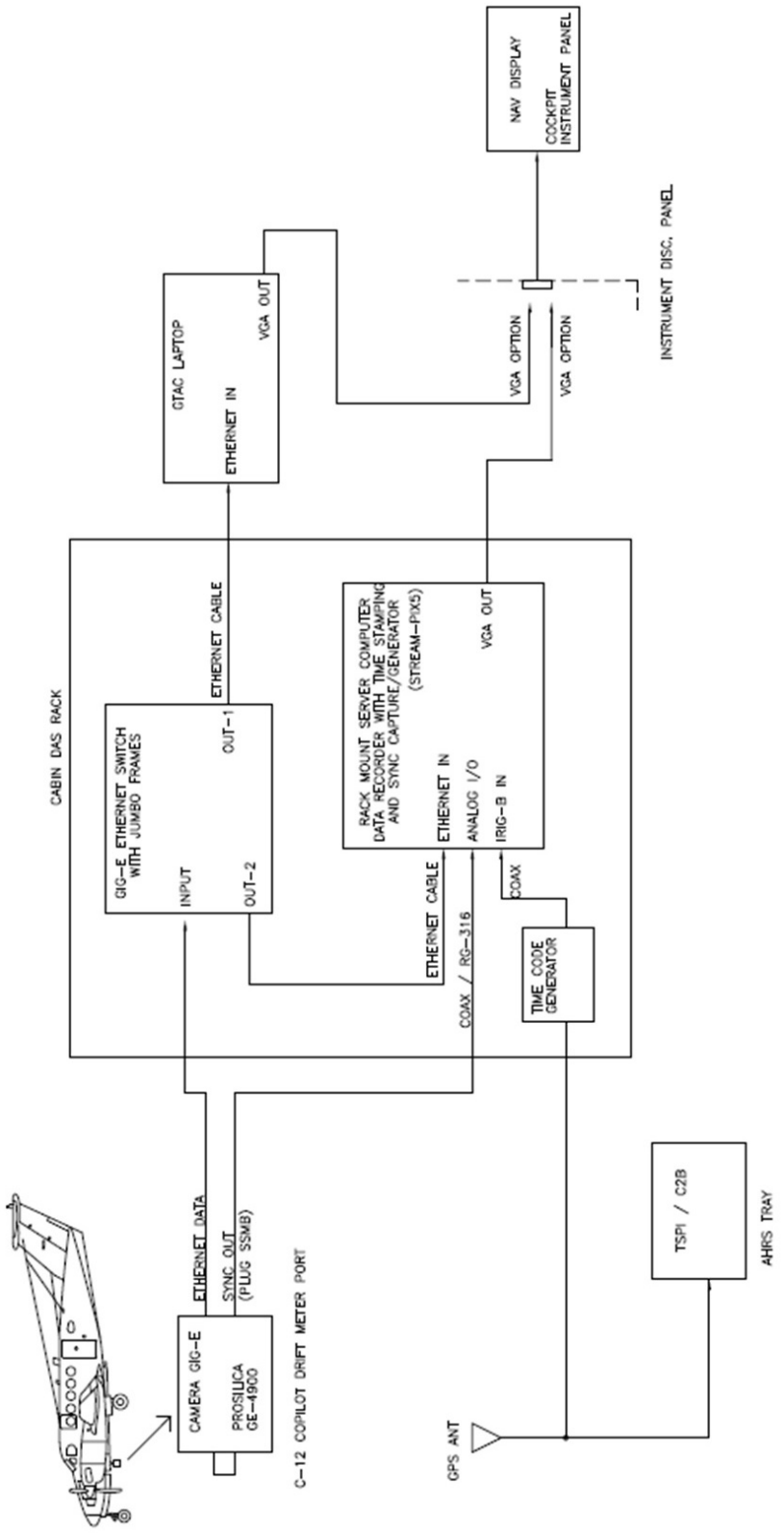


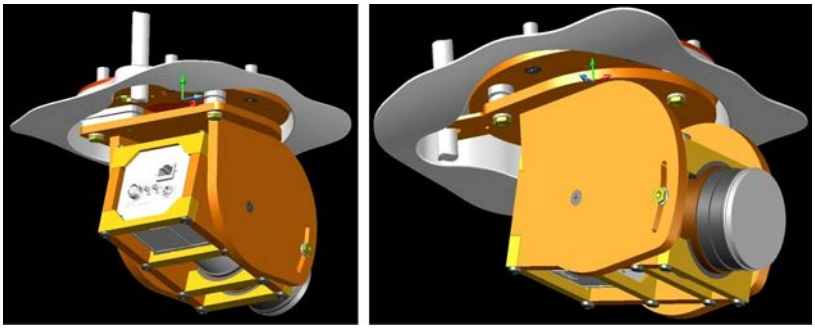
Figure 4.9: Data collection system hardware schematic. All hardware shown was installed on the C-12C test aircraft.

Figure 4.10: C-12C "Huron" aircraft. The relatively spacious cabin and cruising speed of 120-150 knots made the Huron an ideal data collection platform.

vertical. The camera was capable of transferring three uncompressed frames per second (fps) via an ethernet connection. The camera was mounted sideways and at a 12.5° downward angle under the aircraft to maximize the vertical view. Complete technical specifications are provided in Appendix A.

The camera lens was a 50 millimeter focal length Zeiss Planar T* 1,4/50 ZF, part number 15670459. Its manual aperture setting was intended to be set to full-closed and focus set on infinity during all flight testing. More information on the lens is provided in Appendix A.

The camera was mounted in a custom heavy-duty metal enclosure through a driftmeter port in the floor of the cabin under the copilot's seat as shown in Figure 4.11. Not all C-12C aircraft have a driftmeter port; this aircraft was modified during a previous flight test program. Since the test project was not given exclusive use of the aircraft during the flight test period and the camera would have to be removed for other flights in the aircraft, it was highly desired to have a means of reinstalling the camera without requiring the re-boresighting of the camera. The chosen design held the camera in place with precision-machined grooves and four machine bolts to a tolerance of 0.002 inches.



(a) CAD depiction of camera mount.

(b) Camera mounted through driftmeter port under copilot on C-12C.

Figure 4.11: Camera mount installed on test aircraft. The mount is designed to maintain a ± 0.002 inch tolerance between installations.

4.2.3 On-board Computers and Network Switch. Two computers, as shown in Figure 4.12 were carried onboard for data collection and control of the camera. The primary computational duties were performed by a tower desktop computer located in the equipment rack. This computer had an Intel Xeon 64-bit LGA-771 quad-core processor with a front bus speed of 1333 megahertz (MHz), 8 gigabytes (GB) of Random Access Memory (RAM), and a 500 GB hard drive. The computer controlled the camera through a network switch via a 10 gigabit (10 gigabit per second transfer rate) ethernet connection. A Getac B300 ruggedized laptop with a 2.0 gigahertz (GHz) Intel i7 processor, 4 GB RAM, and a 10-gigabit ethernet port was used to control the AMPEX computer via a Windows Remote Desktop connection. The network switch made simultaneous ethernet traffic possible. After each flight, all imagery data was transferred to a one terabyte (TB) portable hard drive. This process took from one to two hours and required the aircraft to remain on ground power.

Figure 4.12: Installed equipment rack with tower computer beneath and laptop velcroed on top. The laptop controlled the rest of the hardware through a remote desktop connection.

4.2.4 StreamPix5® Software, Time-Code Generator, and Pilot Display. StreamPix5, sold by Norpix Incorporated, is a specialized program designed to control scientific cameras and provide real-time digital video recording to the hard drive. The real-time video captured during flight test was stored to the AMPEX hard drive in Tagged Image File Format (TIFF). The camera was allowed to capture images at the maximum rate it was capable (averaging three frames per second) and sent a trigger to the computer, notifying StreamPix5, when an image was captured. A hardware TCG signal was input to a special interface card in the computer to synch it to actual GPS time (translated to local 24-hour time) and to time-stamp the image file names with this time. The StreamPix5 video signal was also sent to the pilots' center MFD for real-time viewing of the camera images to aid in target tracking.

Figure 4.13: Pilot display showing real-time camera view. This modification allowed the pilots to accurately fly the aircraft to collect the required visual data.

4.2.5 GLite. A Configuration 2B GPS-Aided Inertial Navigation Reference Unit (GAINR) Lite was used to record raw inertial measurements, real-time GPS-aided Kalman filter data, and additional information needed for the post-processed TSPI data. The GLite contained a Honeywell HG1700 Inertial Measurement Unit (IMU), a Personal Computer Memory Card International Association (PCMCIA) slot to hold the removable data card, a GPS receiver, an internal computer and

software, and various interfaces [24]. The HG1700 IMU is the same type found in JDAM weapons. After each flight, the PCMCIA card contents were copied to the author's computer before TSPI processing which typically took about a week. The GPS receiver provided raw measurements of L1 C/A code pseudorange and carrier phase and non-differential instantaneous position estimates at 10 samples per second and blended to create the real-time solution. The raw IMU measurements were taken at 100 samples per second.

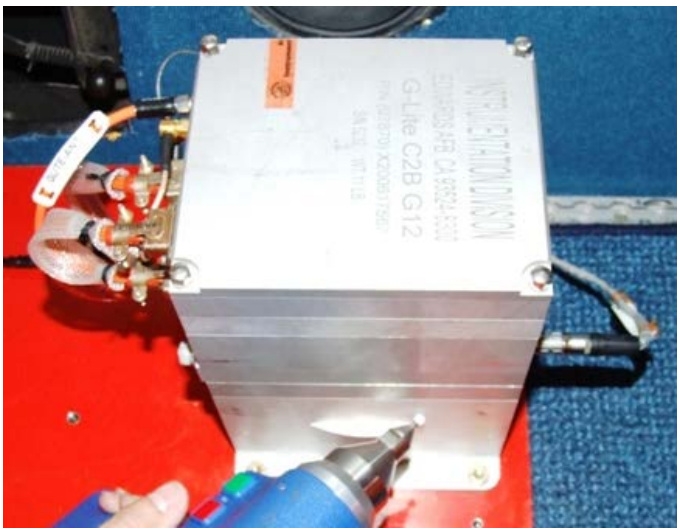


Figure 4.14: GLite precision navigation unit. This hardware recorded raw inertial, a real-time GPS-aided Kalman filter position solution, and additional data to create highly-accurate post-processed TSPI data.

4.2.6 Data Processing Computer. Algorithm coding and post-flight data processing were performed on an Apple MacBook Pro with a 2.67 gigahertz Intel quad-core i7 CPU, 8 GB of RAM, a 500 GB hard drive, and a NVIDIA GeForce GT330M graphics card with 512 MB of video RAM, running the 64-bit version of Windows 7 Home Premium Edition.

4.2.7 Camera for Texture-Mapping Three-Dimensional Models. A small "point and shoot" camera was used to take pictures of the objects to modeled in the target environment. The camera used was a 14.7 MP Samsung TL34HD.

4.2.8 Google Sketchup Pro® Software. To construct the virtual worlds required by the VANSPR algorithm, the COTS Google Sketchup Pro® software was used. This Computer Aided Design (CAD) software package allowed the user to quickly create representative shapes of all objects of interest in the target environments and texture-map photographs of the corresponding surfaces to the target shapes. Target environments were created by building individual component shapes to scale separately, followed by combining them into a master file. The ground surfaces for the environments were created by texture-mapping high-resolution aerial imagery on flat surfaces. The free version of the software (Google Sketchup) could be used for all object model creation, but the commercial version (Google Sketchup Pro®) was required to export the models to VRML format for use in MATLAB®.

4.2.9 MATLAB® Version R2010b Software. MATLAB® is a computing environment and programming language commonly used in engineering applications. Program execution is not particularly fast, but the language is relatively easy to program, debug, and modify for initial algorithm development. Additionally, many "toolboxes" have been developed to speed development in specific research areas. The VANSPR algorithm, written in MATLAB®, uses the 3D Simulink Animation Toolbox (also known as Virtual Reality Toolbox in older versions) for VRML model control. The MathWorks Inc. generally releases two version each year with new features and bug fixes; certain functions of the VANSPR algorithm require version 2008b or later. The 64-bit Windows version was required for some phases of the analysis due to the extra accessible memory locations.

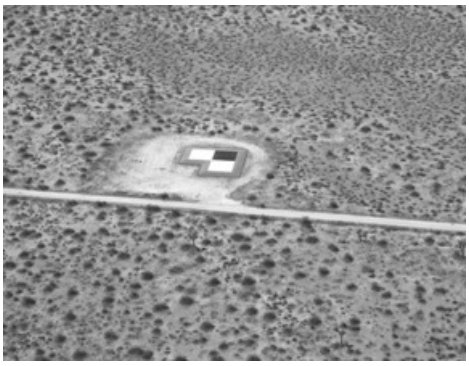
4.2.10 Air Force Flight Test Center Resources. All flight testing was accomplished on the Precision Impact Range Area (PIRA) located at Edwards Air Force Base (AFB), California. The Air Force Flight Test Center (AFFTC) Range Division maintains differential GPS antennas and provided post-processing on the GLite recorded data to create the highly-accurate TSPI data.

Five locations in the PIRA were used as targets during the test flights including the Solar Active Edge Corner (SAEC) board, a T-43 tank, a conex structure, Cowbell Tower, and the X-33 compound, as shown in Figure 4.15. The locations of the chosen targets in the PIRA are shown in Figure 4.16. These five targets were chosen because of their variety of terrain, shape, size, number of objects, and contrast. Only existing targets were used; nothing new was created.

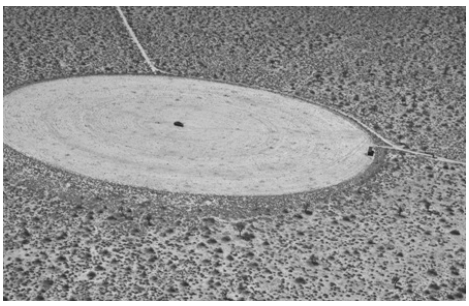
4.3 Flight Test Planning and Execution

There are an infinite number of variables associated with target environments such as shape, height, weather, approach to the target, contrast, etc. The three main factors of interest used in a Design of Experiments (DOE) flight test design were target type, run-in angle, and flight profile. The primary considerations for the target environments were location in consideration of limited flight time, shape, number of structures in the target environment, and contrast. The five main target environments were the X-33 compound, Cowbell Tower, SAEC, tank on PB-9, and a conex structure. These target environments possessed varying degrees of the proposed considerations. In addition to target considerations, the VANSPR algorithm was also challenged by run-in angle and approach to target variations. Flight condition variables including shadows, specific pilot, and time of day were minimized to the maximum extent possible. These variables were minimized by flying the sorties close to a time of the day when shadows were minimal, flying at close to the same time each day, and flying repeatable flight test techniques.

Sixty data collection runs were conducted over the course of eight test flights against five different target environments using two different attack profiles. The cruise profile, explained in greater depth in the next subsection, was flown at altitudes of 200, 500, and 2000 feet AGL and at three different run-in headings for each target. Table 4.1 shows the cruise profile test points and Table 4.2 shows the weapon profile test points. Figures 4.17, 4.18, 4.19, 4.20, and 4.21 depict the various run-in headings flown against each target. The X-33 compound necessitated modified run-in headings



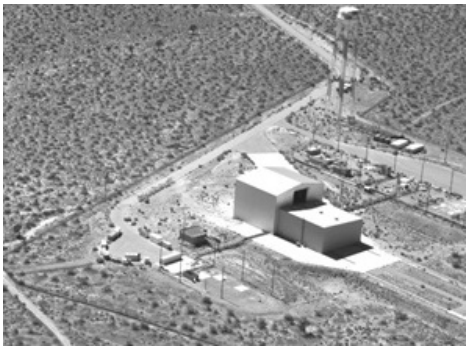
(a) SAEC target environment. This was the only flat target used.



(b) PB-9 Tank target environment. A shipping container is also located on the edge of the circle.

(c) Cowbell Tower target environment. This area includes a tank, toilet, construction vehicle, and containers.

(d) Conex target environment. This environment also included four airplanes and a petroleum truck.



(e) X-33 target environment. This complex included over 40 separate target objects.

Figure 4.15: Selected PIRA target environments.

Figure 4.16: Location of targets on PIRA. The top of the photograph is north; the Edwards AFB runways are to the left of the image (not shown). The distance from Cowbell Tower to PB-9 is 9.2 statue miles.

Figure 4.17: Run-in headings against SAEC board. The magnetic headings shown correspond to 0°, 30°, and 45° from a heading chosen as perpendicular to one side of the board.

for the 200 foot AGL test points in order to safely avoid the 250 foot water tower in the complex.

4.3.1 Cruise Profiles. The cruise profile test runs were accomplished at a constant altitude. The planned altitudes were 200 feet (with a tolerance of +200/-0 feet), 500 ±200 feet, and 2,000 ±200 feet AGL, at a speed of 120 ±10 Knots Indicated Air Speed (KIAS), with flaps set at 40%. A pitch rap or wing rock, depending on the test point, was accomplished by the pilot prior to initiating the data run for the purpose of correlating TSPI and inertial data (from the GLite) in postflight analysis. The data collected on the cruise profiles is not discussed in this thesis but is available for follow-on work. The nature of the data is very similar to that collected from the weapon profile test points but less dynamic and not realistic for what a weapon-mounted camera would see.

4.3.2 Weapon Profile. The weapon profile test runs attempted to approximate the image geometry and dynamics encountered by an optically guided weapon during the terminal phase of flight, within the constraints of aircraft performance and safety of flight. The trajectory, depicted in Figure 4.22, was a wings level dive

Table 4.1: Flight test sortie matrix of cruise profiles. The first number is the sortie the test point was accomplished on. The second number is the run-in heading. Bad data is indicated by "XX". The 0°, 30°, 45° headings refer to the aspect to chosen target surface.

	200 ft AGL			500 ft AGL			2000 ft AGL		
Target	0°	30°	45°	0°	30°	45°	0°	30°	45°
SAEC	5 / 253°	5 / 223°	6 / 208°	6 / 253°	7 / 223°	4 / 208°	2 / 253°	2 / 223°	2 / 208°
Conex	5 / 255°	5 / 225°	5 / 210°	8 / 255°	2 / 225°	2 / 210°	7 / 255°	8 / 225°	7 / 210°
Tank (PB-9)	6 / 270°	5 / 240°	5 / 225°	6 / 270°	6 / 240°	2 / 225°	3 / 270°	3 / 240°	7 / 225°
Cowbell Tower	5 / 255°	5 / 225°	6 / 210°	8 / 255°	6 / 255°	4 / 210°	2 / 255°	4 / 225°	2 / 210°
X-33 compound	6 / 275°	5 / 305°	6 / 320°	6 / 275°	XX	2 / 230°	4 / 275°	8 / 245°	5 / 230°

Table 4.2: Flight test sortie matrix of weapon profiles. The first number is sortie the test point was accomplished on. The second number is the run-in heading. Bad data is indicated by "XX". The 0°, 30°, 45° headings refer to the aspect to chosen target surface.

	Weapon Profile		
Target	0°	30°	45°
SAEC	XX	7 / 223°	3 / 208°
Conex	5 / 255°	5 / 225°	4 / 210°
Tank (PB-9)	4 / 270°	3 / 240°	7 / 225°
Cowbell Tower	4 / 255°	7 / 225°	7 / 210°
X-33 compound	4 / 275°	8 / 245°	7 / 230°

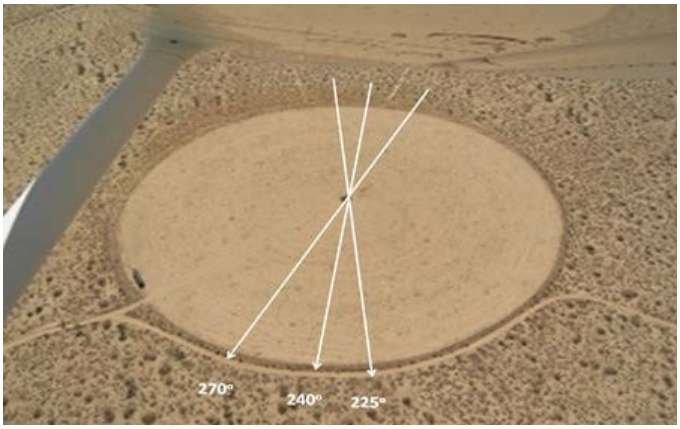


Figure 4.18: Run-in headings against tank on PB-9. The magnetic headings shown correspond to 0°, 30°, and 45° from a heading chosen as perpendicular to the broadside of the tank.

Figure 4.19: Run-in headings against Cowbell Tower. The magnetic headings shown correspond to 0°, 30°, and 45° from a heading chosen as perpendicular to one side of the tower cab.

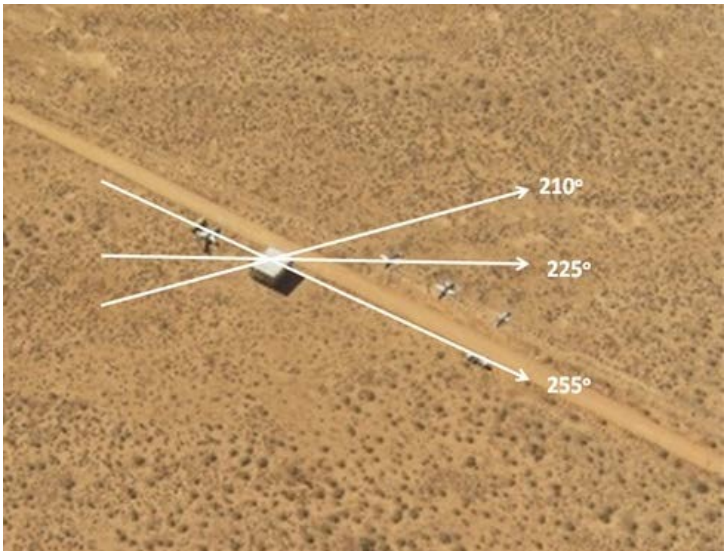


Figure 4.20: Run-in headings against conex structure. The magnetic headings shown correspond to 0°, 30°, and 45° from a heading chosen as perpendicular to one side of the conex structure.

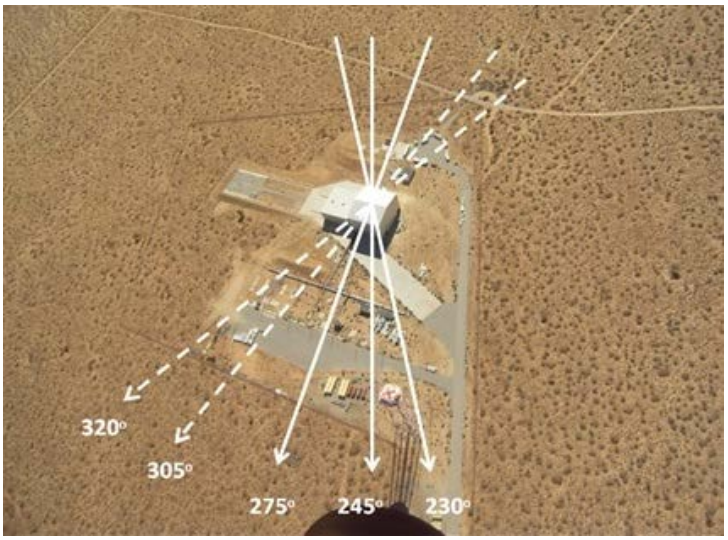


Figure 4.21: Run-in headings against X-33 compound. The magnetic headings shown correspond to 0°, 30°, and 45° from a heading chosen as perpendicular to one side of the largest building. The 200 foot AGL test points required that the 30° and 45° run-ins be altered to provide adequate spacing from the 250 foot water tower.

profile towards a ground target. The points were planned to collect approximately 90 seconds of data with the target in a stable position on a navigation display. A pitch rap was accomplished by the pilot prior to initiating the run for the purpose of correlating TSPI and inertial data (from the GLite).

The maneuver was initiated from level flight at 4,100 ± 200 feet AGL, 160 ± 20 KIAS, and 40% flaps on the specified run-in magnetic course ± 10°. Between 4.9-5.8 nautical miles (nm) from the target, a dive angle between 6-8° was established to put the centroid of target approximately one quarter of the way down from the top of the navigation display. Due to environmental conditions, the pilot adjusted the push over distance and/or initiation altitude in order to achieve the desired dive angle. The centroid of the target was maintained at this position for the entire dive. Recovery was initiated no later than 500 feet AGL for final dive angles of less than or equal to 10° and 800 ft AGL for final dive angles between 10° and 15°. The planned maximum altitude lost during the dive recovery was 73 feet. Therefore, the minimum altitude during the dive recovery was planned to be 427 feet AGL. The recovery and abort

procedures were for both to simultaneously add full power, pull to and maintain 1.8g, and maintain constant airspeed until a positive rate of climb was established.

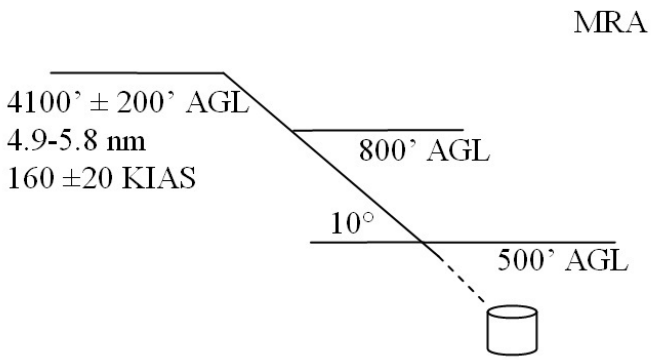


Figure 4.22: Weapon profile. The data collection run begins at 4,100 feet AGL and transitions to a 10° dive at approximately 5 nm from the target. Recovery was initiated at 500 feet AGL.

4.3.3 Flight Test Time Alignment Procedures. Proper alignment of time-stamped data from multiple sources is critical in an estimation algorithm. A mismatch could lead to time correlation issues - a violation of Kalman filter assumptions. To help ensure all recorded data was being properly time-stamped, each data collection run was initiated with a pitch rap or wing rock. In post-flight analysis, these maneuvers were observable in the visual, inertial, TSPI, and real-time Kalman filtered data sets. It was determined that the wing rock was the more desirable maneuver in that it was easy to find in the vast data because of its visual significance (easy to see rocking in a sequence of images), longer duration, and the presence of multiple distinct points (maximum bank angle in both directions, passing through or returning to wings level flight).

The 412 Range Squadron provided a simple C^{++} software script that read the real-time GPS-aided Kalman filter navigation solution file and converted it to a Google Earth compatible data file. This allowed the test team to copy the data off the GLite data card before turning it in for processing (which typically took about seven days) and check for reasonableness. Figure 4.23 shows the complete trajectory of a test

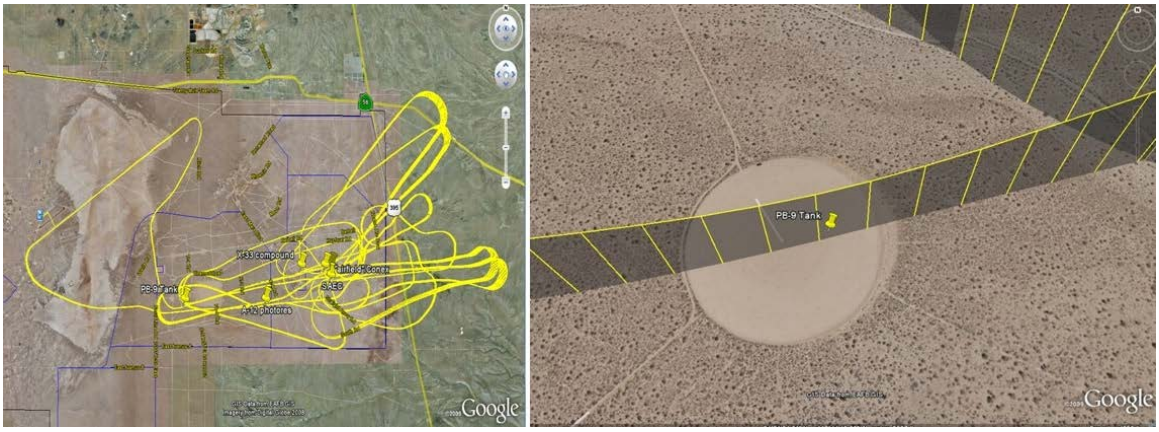


Figure 4.23: Three-dimensional real-time position information plotted in Google Earth. This data, 99.4% similar to TSPI was available immediately after each test flight.

flight in Google Earth. Note that altitude was also recorded and depicted three-dimensionally.

Another method was employed on the ground before several flight to detect any latency between image capture and time-stamping. Images were time-stamped in-flight using the same time code generator signal used by the GLite for time-stamping TSPI and real-time GPS-aided Kalman-filtered data. The time stamp of each image was presented as the file name of the image file. In order to characterize the time-latency (i.e. delay) of the time-stamping of the images, video of a separate, highly-accurate clock was taken. The separate system used the same model of TCG to ensure meaningful results. The time shown in the image was compared to the time-stamp (image file name) for each image. The average time-delay for was shown to be 0.077 seconds with a standard deviation of 0.044 seconds using 163 time-stamped images. The largest time lag was 0.561 seconds and the smallest time lag was 0.036 seconds. Figure 4.24 shows the highly-accurate TCG-driven time display that was filmed with the test camera to see when the images were *really* time-stamped and stored.

4.3.4 Gross Distortion Removal. Three iterations of the Caltech's MATLAB® Camera Calibration Toolkit were conducted to minimize the lens distortion present in the collected data images. The camera calibration software determined the differ-

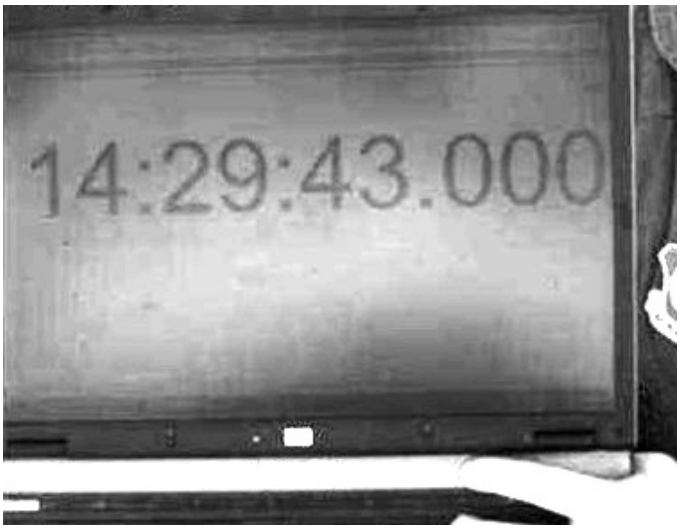


Figure 4.24: Laptop showing highly-accurate time from a connected time code generator (TCG). The laptop screen showing the time was imaged and compared with the time-stamp triggered by the test camera to determine time-stamp latency.

ences between pixel locations of square intersections on the calibration board with where they should have been (since the calibration board object was known to the application). The pixel error was decreased in each of the first three iterations as shown in Table 4.3. The refined corner locations were re-examined after each iteration to ensure that they fell at the square intersections on the calibration board in the calibration images. A fourth iteration yielded incorrect reprojection of the corner points; therefore, the refinement was considered complete after three iterations. The numerical errors were approximately three times the standard deviation in pixels, as defined by the software package.

The five image distortion coefficients, described in Equations (2.90) and (2.91) were calculated. The results are shown in Table 4.4. The pixel error was determined to be 1.41617 pixels horizontally and 1.70991 pixels vertically. Tangential error values were relatively low, accounting for no more than 1.6 pixel shifts to remove distortion. Radial distortion shifts, however, approached 30 pixels of error. Therefore, the total distortion model appears to be almost exclusively comprised of radial distortion.

Table 4.3: Resultant pixel errors from reiterations of distortion removal using the Caltech Camera Calibration Toolbox. The first three iterations resulted in pixel error improvement while a fourth iteration resulted in a greater pixel error.

Iteration	Pixel Error (x)	Pixel Error (y)	Comments
1	1.49866	1.87658	
2	1.42494	1.72067	
3	1.41617	1.70991	Lowest error; optimal iteration
4	1.42493	1.72066	Pixel error getting worse

Table 4.4: Calculated image distortion coefficients. The five values were solved using the Caltech Camera Calibration Toolbox.

Image Distortion Coefficient	Coefficient Value
kc_1	-0.173620
kc_2	0.206130
kc_3	0.001160
kc_4	0.000014
kc_5	0.000000

4.4 Virtual World Model Construction

Three-dimensional models of target objects, accurate to the nearest inch for rectangular targets and to the nearest six inches for objects with highly irregular geometry, were created. The flat surrounding target environments were constructed to an area of at least four square miles for each target using three-inch imagery. The imagery was collected in 2008. The imagery ws validated and determined to be sufficient to meet the test objectives. Height, width, and depth of target objects were collected using hand-held tape measures to the nearest inch. When available, building height data were acquired from the 412 Range Squadron at Edwards AFB. All structures and objects with a height of greater than five feet were measured. Google Sketchup Pro®, the software used to construct the models, allowed for dimensions of objects to be defined to accuracies less than an inch. A Samsung TL34HD camera was used to photograph all target surfaces to include the tops of the targets. Each surface of the objects was photographed at an angle as close to normal to the surface as possible. Physical separation of objects was often less than two feet and prevented pictures being taken of adjacent surfaces. This omission of pictures was not problematic because these surfaces were not visible by the camera during the flight test. Before using these photographs as object textures, the lens image distortion was removed using the same procedure that was used for the Prosilica 4900 test camera. Distortion removal was accomplished to minimize pixel location errors in views of the models when compared to actual images taken with the Prosilica camera. Additionally, the photographs were down sampled using MATLAB® from 4384×3288 pixels to 800×600 pixels and converted to grayscale. Converting the images to grayscale and reducing the resolution alleviated some of the computational burden during later model view re-orientations while maintaining the required quality. Three-inch geo-orthorectified aerial imagery, shown in Figure 4.25, provided by 95ABW/CEV Environmental Management Division was used to build the model environments. Geo-orthorectification is the process of removing parallax due to camera angle and correlating pixels to latitude and longitude. In addition, the images are oriented to true north. These image tiles were

then pieced together in the CAD software before target objects were added as shown in Figure 4.26.

(a) Imagery tile to the north of the SAEC target.

(b) Imagery tile containing the SAEC target.

Figure 4.25: Geo-orthorectified three-inch resolution overhead imagery tile (2,296.5 feet × 3,609 feet). PIRA target environments were built using these high-resolution image tiles.

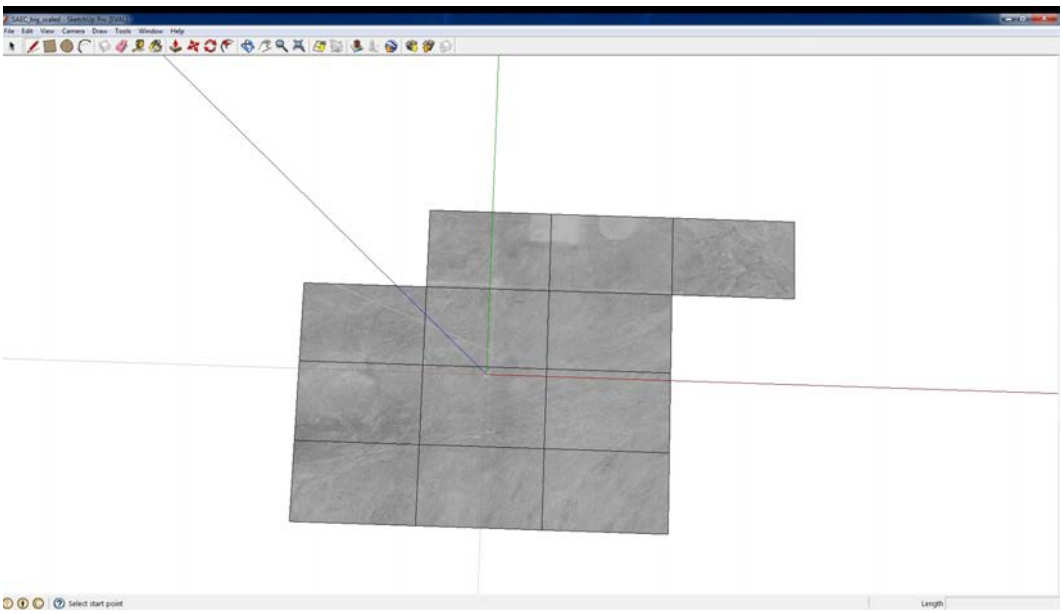


Figure 4.26: Assembly of multiple imagery tiles in Google Sketchup Pro®. Target objects and structures may be added after the flat ground environment is created.

Particular attention should be paid to the units in MATLAB® and those of Google Sketchup Pro®. Google Sketchup Pro® allows for use of English or metric units in construction but MATLAB® does not. In VRML as interpreted by MATLAB®, one "unit" is equal to 40 meters. Therefore, Google Sketchup Pro® models must be shrunk by a 40:1 ratio before exporting to VRML format. This can be easily done after the complete model is built. If the model is not shrunk before use in MATLAB®, a z-frame clipping issue will prevent viewing the model even at reasonable distances.

Three-dimensional target objects *within* a target environment were individually created and then imported into the environment scene. Figure 4.27 is an example of a created object, before and after texture-mapping has been applied. Figure 4.28 shows a more complex target environment object that was created. The VANSPR algorithm increases accuracy with closing range to targets to an even greater degree than possible with other methods due the scalability of the realized resolution.

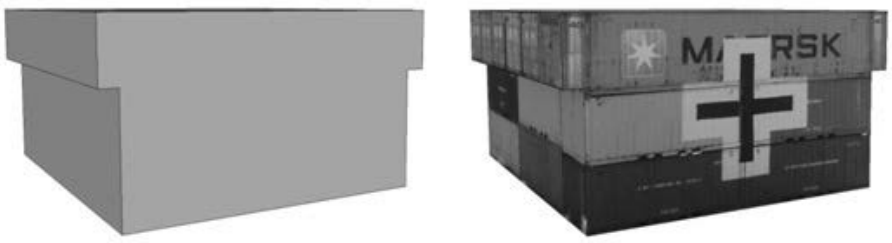


Figure 4.27: Untextured and textured instances of the conex structure target object. Textures can add a high degree of realism to the synthetic object.

The remainder of this chapter provides further graphical examples of target scenes that have been virtually recreated in the CAD software. Figures 4.29, 4.30, 4.31, 4.32, and 4.33 show an actual image from flight test followed by three virtual viewpoints made possible with the created virtual world. An example of a weapon trajectory to a target, that was not possible during the flight test data collection, is shown in Figure 4.34 to illustrate the possibilities of the method.

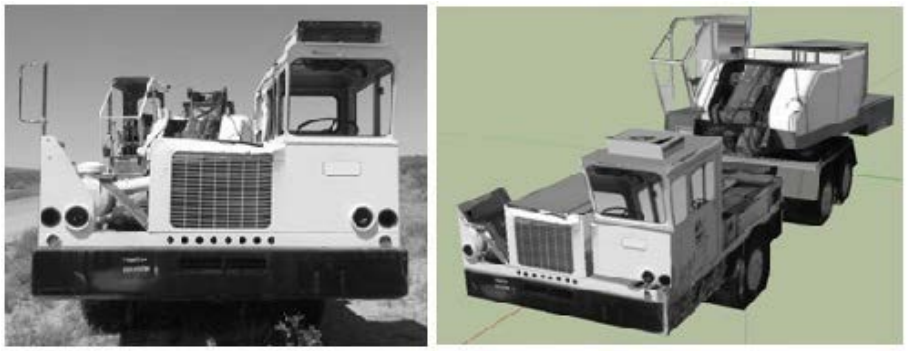
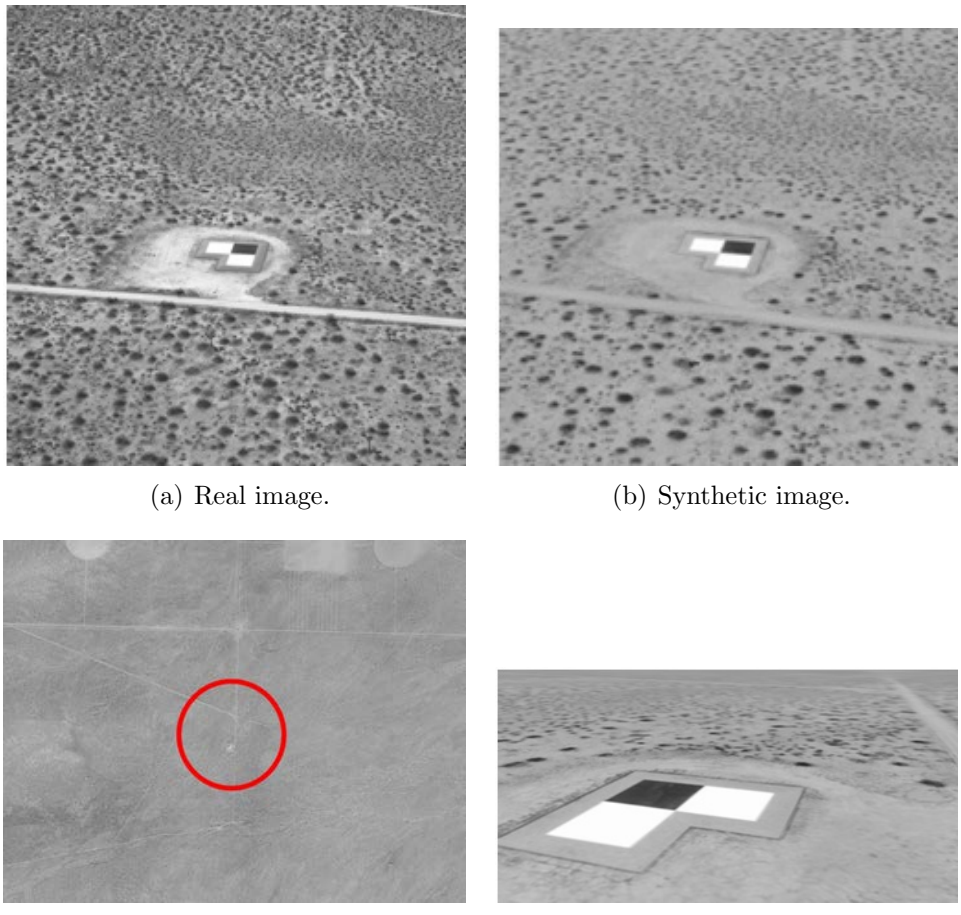


Figure 4.28: Actual photograph of crane vehicle (left) and computer model (right). The computer generated model allows for any viewpoint of the crane to be observed.



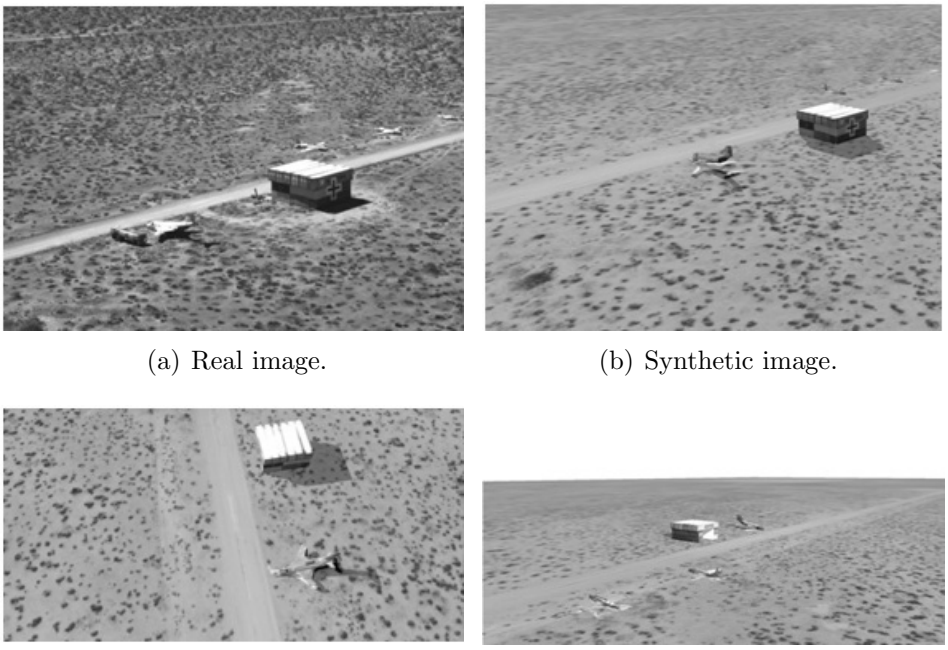
(a) Real image.

(b) Synthetic image.

(c) Synthetic view.

(d) Synthetic view.

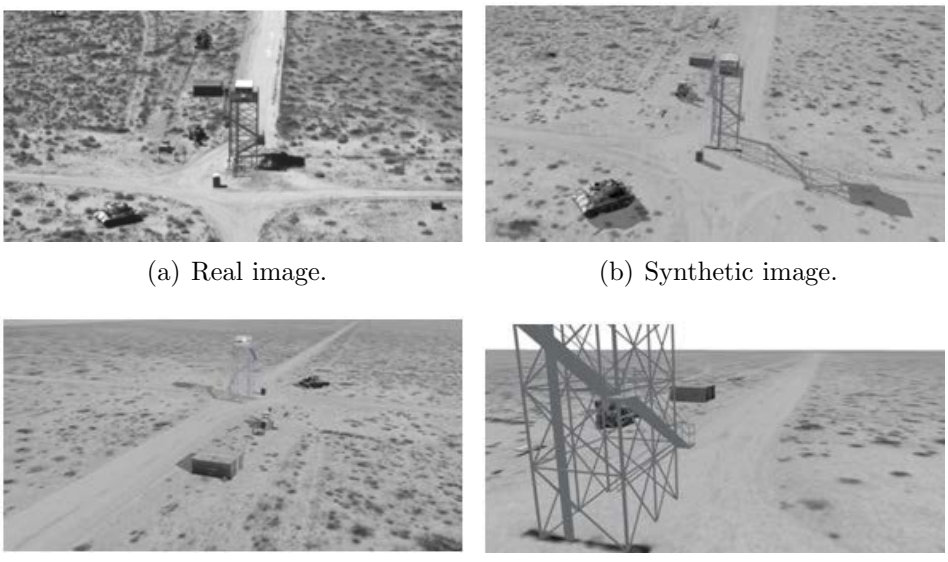
Figure 4.29: SAEC target environment real and synthetic images. This was the only target that had no three-dimensional structure.



(a) Real image. (b) Synthetic image.

(c) Synthetic view. (d) Synthetic view.

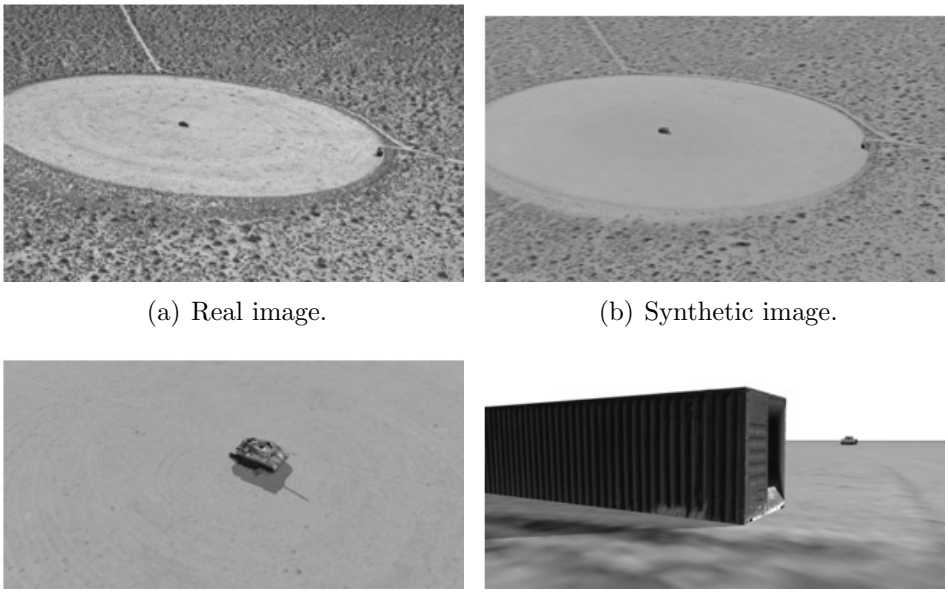
Figure 4.30: Conex target environment real and synthetic images. Synthetic views are shown in Google Sketchup Pro® with shadowing turned on; this is not used in the VANSPR algorithm.



(a) Real image. (b) Synthetic image.

(c) Synthetic view. (d) Synthetic view.

Figure 4.31: Cowbell Tower target environment real and synthetic images. Synthetic views are shown in Google Sketchup Pro® with shadowing turned on; this is not used in the VANSPR algorithm.



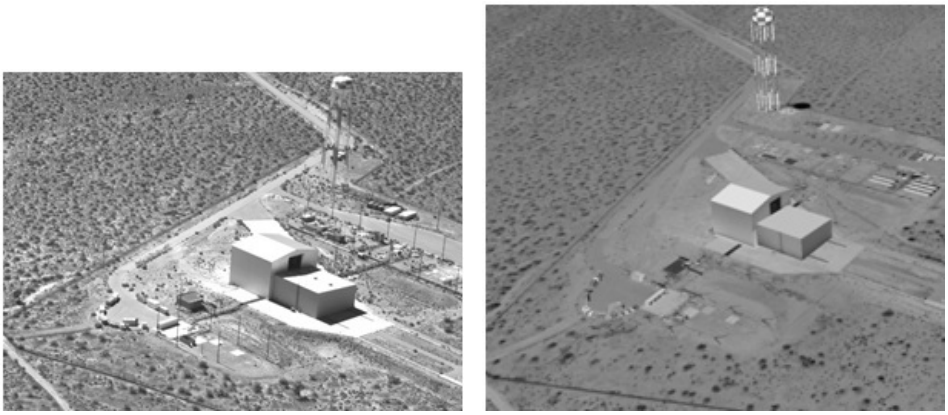
(a) Real image.

(b) Synthetic image.

(c) Synthetic view.

(d) Synthetic view.

Figure 4.32: Tank on PB-9 target environment real and synthetic images Synthetic views are shown in Google Sketchup Pro® with shadowing turned on; this is not used in the VANSPR algorithm.



(a) Real image.

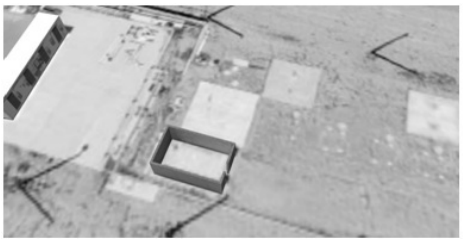
(b) Synthetic image.

(c) Synthetic view.

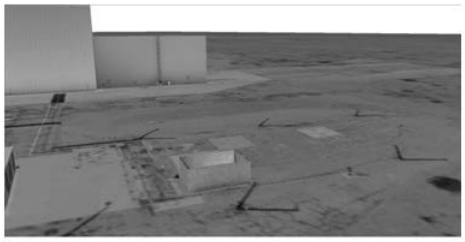
(d) Synthetic view.

Figure 4.33: X-33 compound target environment real and synthetic images. X-33 compound target environment real and synthetic images. Consisting of 42 target objects, this was the largest and most complex target environment.

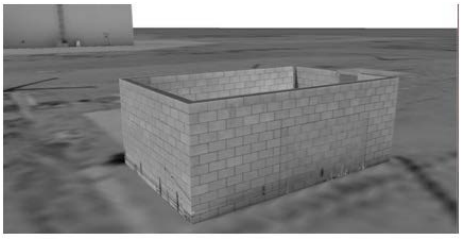
(a) Synthetic view of X-33 target at high altitude.



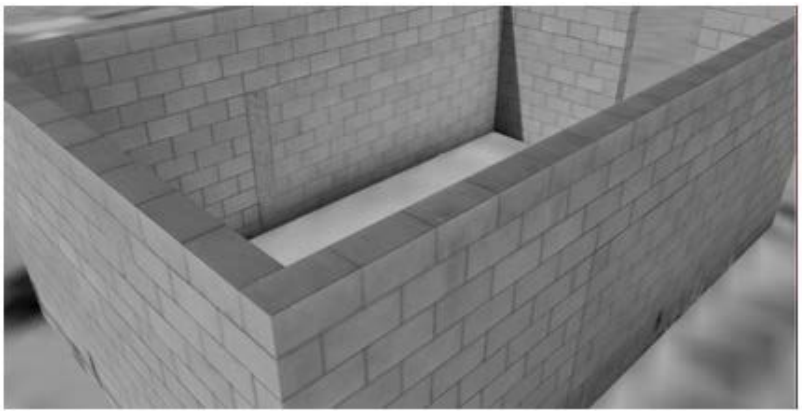
(b) Closer synthetic view of target



(c) Closer synthetic view of target with changed perspective.



(d) Closer synthetic view of target with flatter aspect.



(e) Target synthetic view briefly before simulated impact.

Figure 4.34: Sequence of rendered images as target object is approached. Available resolution to algorithm can be maintained if modeled with sufficiently high-resolution texture maps.

V. Flight Test Experimental Results

This chapter presents performance results of both the underlying algorithms necessary for image correlation and the overall performance of the VANSPR algorithm. A thorough exploration of what *conditioning* had to be performed on the data before it could be used in the algorithm was critical to any success it could have. This included verifying that the virtual worlds were built correctly in that they faithfully reproduced a scene when given truth position and attitude information and verifying that the discussed image processing techniques would be useable on the collected real and synthetic images. After basic success of the VANSPR algorithm was declared, dissection of the data was performed to determine its sensitivity to target type, target aspect, and attack profile. Further analysis was performed to test the algorithm's sensitivity to using three-dimensional models versus flat, shadowing and overexposure (unintended results of Test Sortie 1), image update rate, and image resolution.

5.1 Overview of Collected Data

A large amount of data was collected during the data collection flight test phase of the research. Most of this was the high-resolution images; each uncompressed TIFF format frame was approximately 15 MB. Totals, by data type, are listed in Table 5.1. Data download from the C-12C's installed computer to a portable hard drive took approximately two hours after each sortie.

5.2 Verification of VRML Model

Once all flight test data (to include TSPI "truth", real-time GPS-aided Kalman filtered, inertial, and high-resolution images) were collected, a verification was performed to ensure synthetic views accurately matched real images when built using truth data. In other words, TSPI position and attitude data were fed into the synthetic image generation portion of the VANSPR algorithm to see if the resultant synthetic images looked liked the captured images. The initial comparison using just TSPI data and the reference frame angular rotations between the GLite and test

Table 5.1: Collected flight test data. Each test flight collected high-resolution images, TSPI, and raw inertial data.

Flight	Image Count	Images (GB)	Raw INS (MB)	TSPI (MB)
1	5,665	71.2	142.0	22.2
2	7,627	93.2	182.0	28.0
3	5,135	70.0	143.0	26.6
4	9,189	118.0	140.0	30.8
5	10,175	127.0	215.0	33.6
6	10,607	134.0	215.0	32.4
7	8,281	106.0	152.0	34.0
8	5,666	75.1	98.8	15.3
Totals	**62,275**	**794.5GB**	**222.9MB**	**1287.8MB**
Grand Total	**62,275**	**796.0GB**		

camera provided reasonable results but seemed to show consistent biases. The first goal was to determine and remove those biases.

5.2.1 Determination of Fixed Angle Biases. The original angular rotations between the GLite and camera determined from boresighting were -0.43° yaw, -12.19° of pitch, and -0.28° of roll. By an experimental iterative process of adding biases to each of these angles and comparing the real and synthetic matching on approximately 10 frames on runs spanning all five targets, the following angular biases were determined: -0.3° yaw, -0.1° pitch, and -0.7° roll. While these biases may seem very small and possibly inconsequential, the reader should be reminded that a camera is essentially an angle-detection device and that pixel error increases with distance for an angular error. Therefore, accounting for these small angular anomalies led to much more consistent and repeatable image matching.

5.2.2 Verification of Time-stamp Latency. A method of determining latency from image time-stamping was described in Section 4.3.3. Screen shots of a high-precision clock were filmed by the test camera. The differences between these screen-

captured times and the image time stamps (via the image filename) were compared and analyzed. Performing statistical analysis on 163 images revealed a 0.077 ± 0.044 second one-sigma lag on the time stamps. A correction for this was initially applied to the data; however, later manual comparisons of real and synthetic side-by-side video revealed this adjustment created a greater misalignment. It can be concluded that the time-stamp latency in-flight was different (shorter), but this cannot be quantified since the screen capture method could not be accomplished in flight. Removing the correction appeared to provide the best results; hence, no correction for time latency was applied.

5.2.3 Image Error Due to Constant Elevation Assumption. The virtual ground plane, made up of flat rectangles texture-mapped with aerial-view image tiles as discussed in Section 4.4, does not take into account changing terrain elevation in the target environment. The desire for simplicity was the main motivator. Even with flat terrain, the X-33 compound VRML file was over 100MB, a moderate strain for even today's most elite personal computers. The result of this simplification is visible image mismatching, even when using TSPI "truth" data to build synthetic views at locations other than the central target object which defines the v-frame origin. In general, the mismatch gets worse as distance increases away from the origin as the terrain diverges from the central target object's elevation. Figures 5.1, 5.2, 5.3, and 5.4 give four examples of this effect. Figures 5.1 and 5.3 show a greater mismatch, most visible in the road patterns, since they are a greater distance from the intended target. Figures 5.2 and 5.4 show much better matching since the model and true elevation are close to being equal. For endgame navigation, which is of primary interest in a weapon, this is not a large problem. For enroute navigation, however, this is a bit more troublesome. It would not be wise to make the covariance matrix values ($\mathbf{P}_{xx}$) artificially large and cause the filter to heavily weight an update that would almost surely be incorrect. A more reasonable approach would be to increase the values in

the **R** matrix when less feature matches are available (typically at further ranges), causing the filter to weight the measurement less.

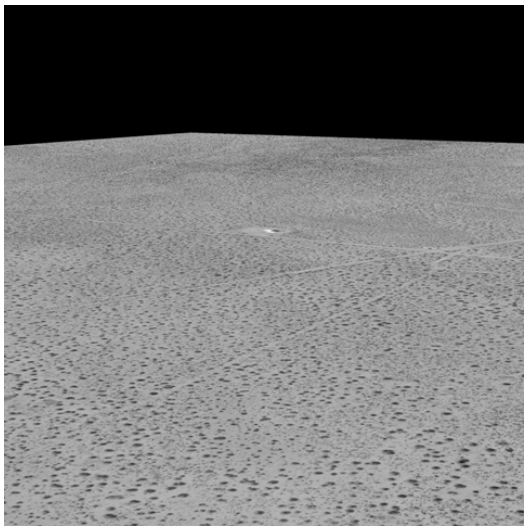
5.3 Verification of Image Comparison Methods

All previously-discussed pixel-based methods, Harris-Stephens corner detection, Hough line detection, and SIFT® matching were examined for viability in comparing real and synthetic images collected and derived from flight test. With the exception of SIFT® matching, which improved in the flight test environment, results were considerably worse. In fact, SIFT® matching was determined to be the only realistic option for an algorithm that shouldn't need to be manually adjusted for different environments and conditions at this stage of development.

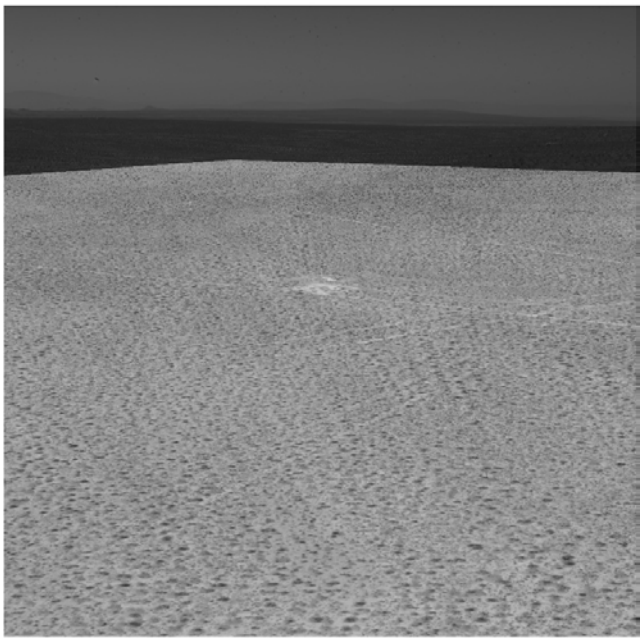
5.3.1 Pixel-based methods. The pixel-based image comparison techniques (SAD, ZSAD, LSAD, SSD, ZSSD, LSSD, and NCC) were applied using three different methods. The first method was to threshold the image, creating a binary image (only black or white pixels), and then apply a pixel-based equation. Approximately 60 combinations of thresholds were applied to both real and synthetic images with no discernable matching pixel patterns. The pixel-based algorithms, therefore, performed as would be expected and picked correct matches 10% of the time. The second method was to first apply a Canny edge detector, which also uses thresholding as a subroutine in its algorithm, before the comparisons. The results were almost identical. Finally, a publicly-available version of the algorithm that processed the intensities of the grayscale (as opposed to binary) image was used. The results were slightly better with about 35% correct matching, but still considered a failure. Applying image-processing techniques such as *sharpening* the image improved results slightly. Again, however, the purpose of the test algorithm is to be autonomous and not require special-case alterations depending on target environment characteristics.

The reason this genre of method worked for Weaver's work is that a single aircraft against a blank background (sky) is a far simpler image processing problem.

(a) Real image.

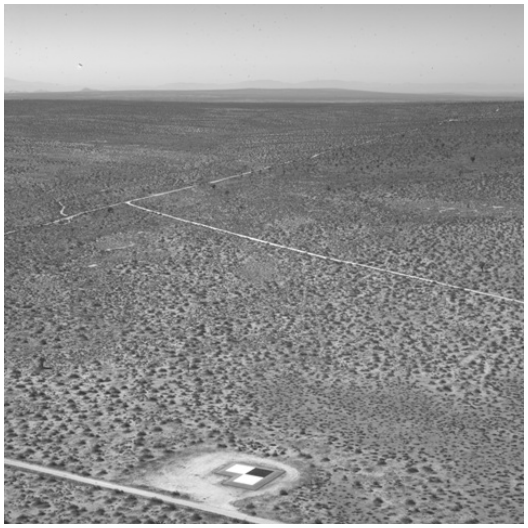


(b) Synthetic image.

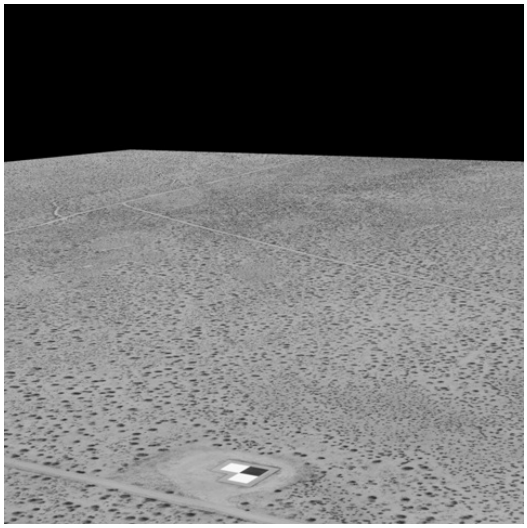


(c) Synthetic image overlaid on real image.

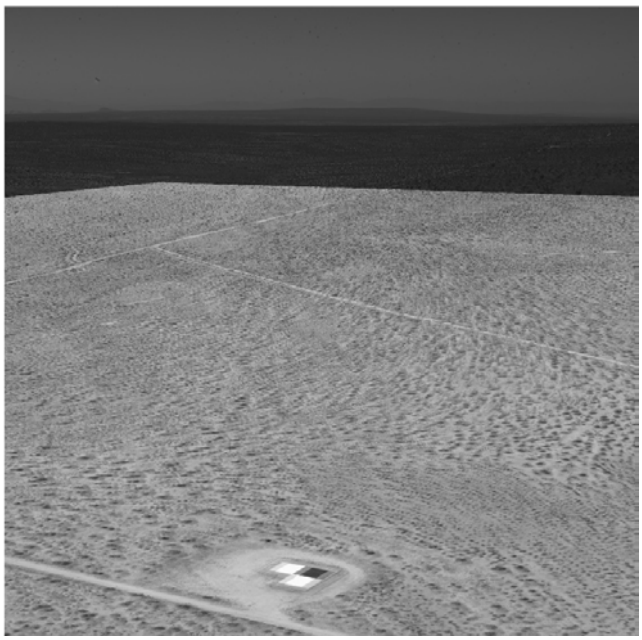
Figure 5.1: Comparison of real and synthetic views early in a run against the SAEC target. The synthetic view was generated with TSPI truth data. Discrepancies in image alignment are primarily due to elevation differences between target and surrounding landscape since the entire surface was modeled as flat with a constant elevation equal to that of the SAEC board.



(a) Real image.

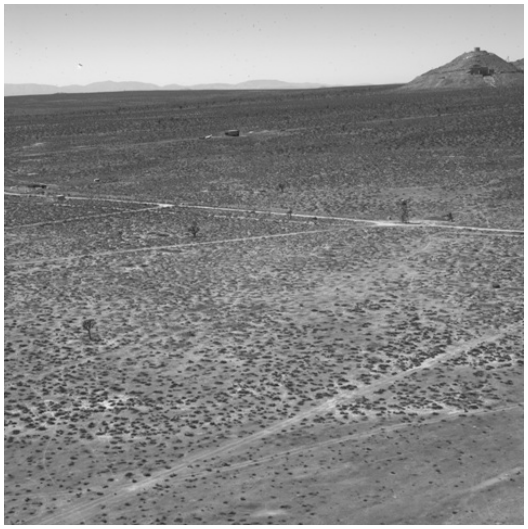


(b) Synthetic image.

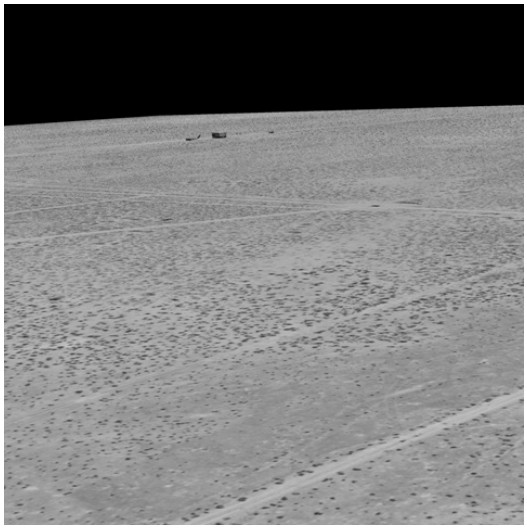


(c) Synthetic image overlaid on real image.

Figure 5.2: Comparison of real and synthetic views late in a run against the SAEC target. The synthetic view was generated with TSPI truth data. Discrepancies in image alignment are small close to the target where elevation deviations are minimal.



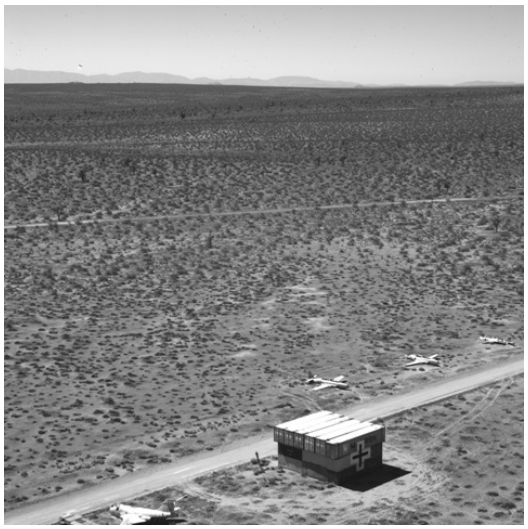
(a) Real image.



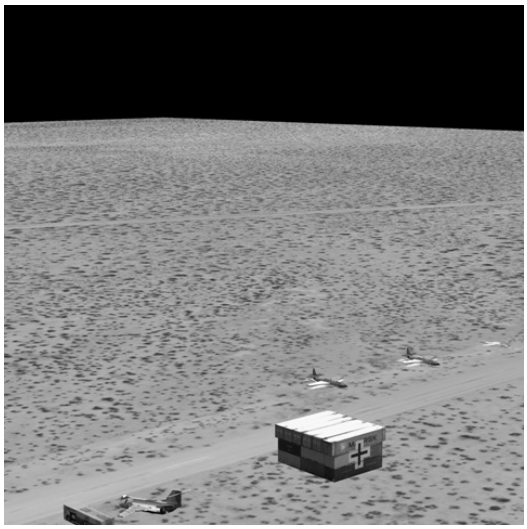
(b) Synthetic image.

(c) Synthetic image overlaid on real image.

Figure 5.3: Comparison of real and synthetic views early in a run against the conex target. The synthetic view was generated with TSPI truth data. Discrepancies in image alignment are primarily due to elevation differences between target and surrounding landscape since the entire surface was modeled as flat with elevation equal to that of the immediate conex structure area.



(a) Real image.



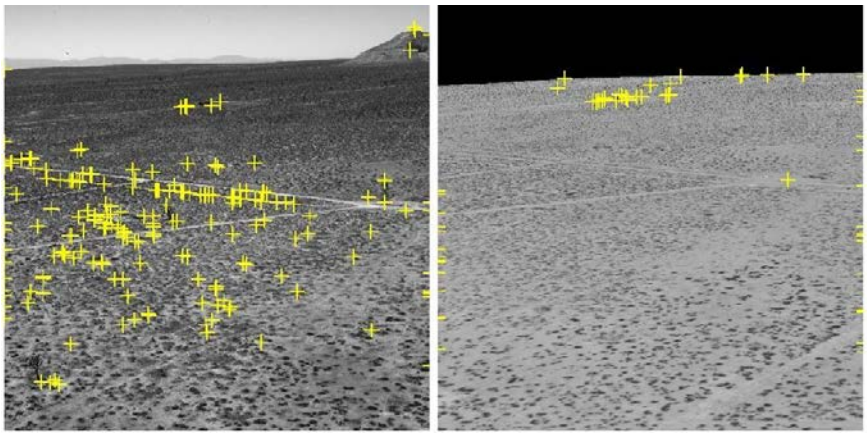
(b) Synthetic image.

(c) Synthetic image overlaid on real image.

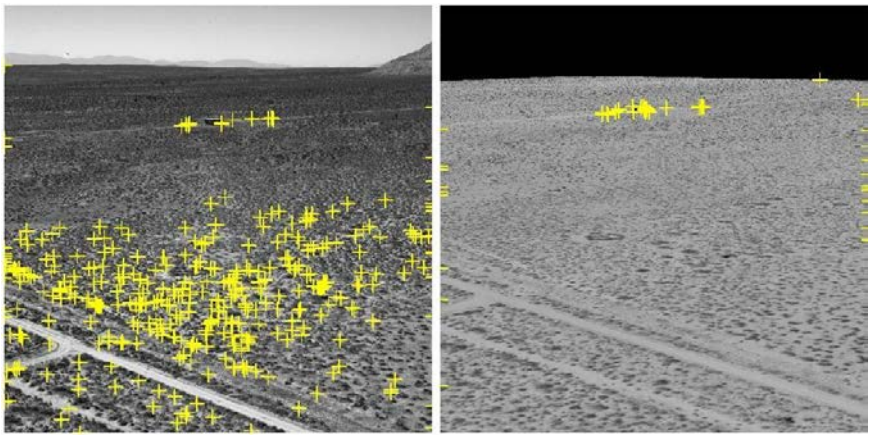
Figure 5.4: Comparison of real and synthetic views late in a run against the conex structure. The synthetic view was generated with TSPI truth data. Discrepancies in image alignment are small close to the target where elevation deviations are minimal.

Detecting the presence of a line, shape, or contrast gradient without any possibility of clutter (clouds and sun glare were avoided) can be handled with a constant parameter algorithm. Ground clutter and changing objects of interest in a downward-looking application presents a different set of challenges that require adaptive pixel-based parameters which was not addressed in this work.

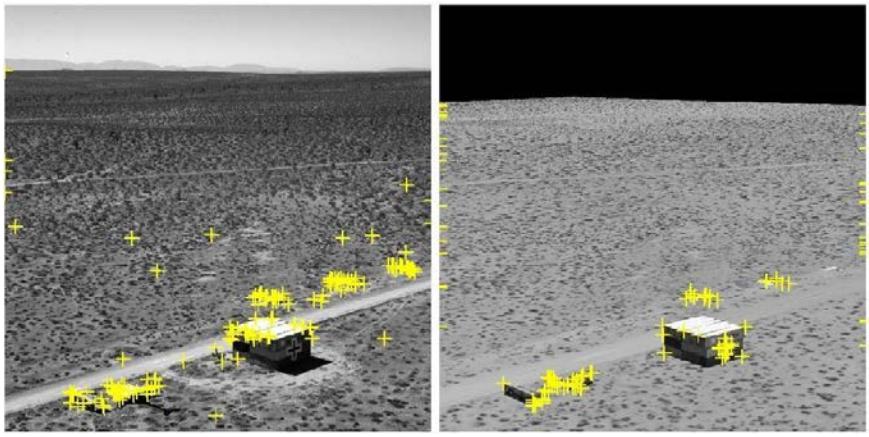
5.3.2 Harris-Stephens Corner Detection. Image comparison using Harris-Stephens Corner Detection was not considered successful. The most obvious discrepancy was the high number of corners found in the real images as compared to the synthetic images when the sensitivity arguments were the same. Adjusting these parameters manually could sometimes yield more corners found in the synthetic view, but then would become worse when applied to a different data set. Without a more robust, automatic method of adjusting these parameters, manual adjustment would be required, negating the desired autonomy of the VANSPR algorithm. Figure 5.5 shows Harris-Stephens corner points on three image pairs taken along a run at the conex target. It can be readily seen that in addition to a much smaller number of corners found in the synthetic image, the points it does find tend to be in the distance and not very well matched with those in the real image. The reason for tending to find points in the distance is primarily due to the artificial edge ("horizon") caused by the boundary of the synthetic world. While simple edges are not detected, lines intersecting the artificial horizon make a corner and are hence detected. In a similar fashion, the simple algorithm finds abundant corners around the edges where lines are running off the image. It should be noted, however, that unlike SIFT®, the Harris-Stephens algorithm as implemented here does not match pairs but simply finds corner points in the images individually. The exception seems to be when very close to objects with very distinctive corners, as shown in Figure 5.5(c). While potentially useable for the last moments of flight for a weapon, the corner detection method was not suitable overall for enroute navigation to the target.



(a) Real (left) and synthetic (right) images early in a conex run with Harris-Stephens corners.



(b) Real (left) and synthetic (right) images midway through a conex run with Harris-Stephens corners.

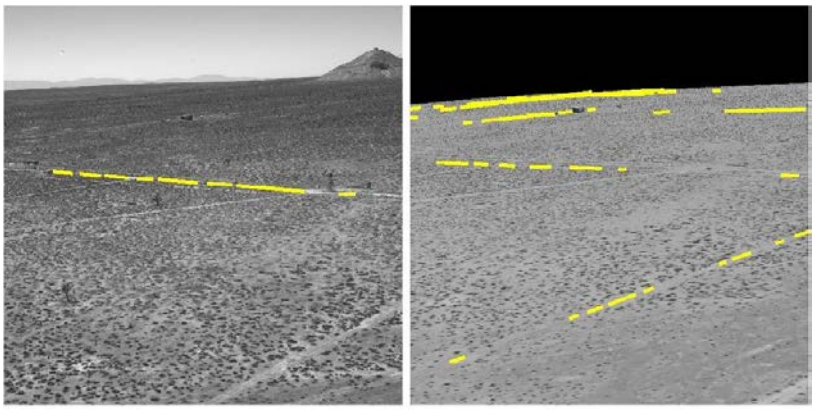


(c) Real (left) and synthetic (right) images late in a conex run with Harris-Stephens corners.

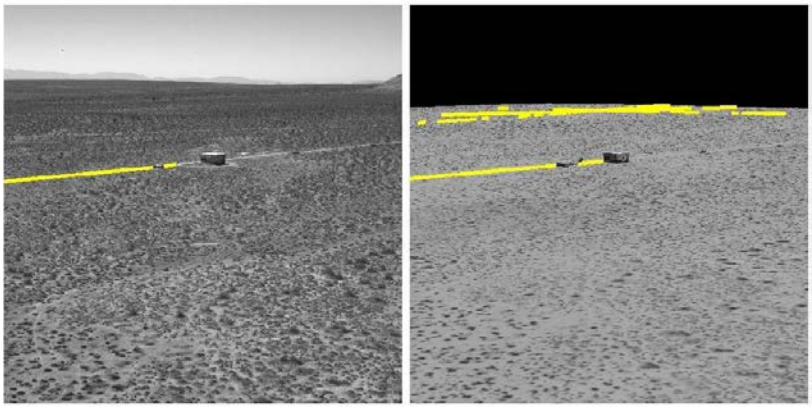
Figure 5.5: Harris-Stephens corner detections in real and synthetic images along a run against the conex structure. The tendency for corners to have greatest density in differing areas in the two image types makes information extraction using this method of minimal value.

5.3.3 Hough Line Detection. The technique of detecting lines in the images by finding peaks of a Hough transform was attempted. The first step was to detect edges in both images using the Canny edge detector algorithm. Through experimentation, the optimal values for obtaining edges in the real image was with hysteresis thresholding limits of 0.3 and 0.55 and a threshold filter standard deviation of one. Optimum results were achieved on the synthetic images using 0.1 and 0.12 for threshold values and also one for the threshold filter standard deviation. After optimum edge detection was achieved in both images independently, the Hough transform was applied to each to find lines. As can be seen from Figure 5.6, correlation of lines for each image pair was poor. Like the Harris-Stephens corner points, the synthetic image tended to find detail and lines in the distance. In the virtual ground plane, since it is really a rotated image of an overhead photograph, features in the distance are not degraded in the same way they would be in a real image. Features are not faded or distorted in the atmosphere, but only compressed due to the optical geometry. Therefore, any high contrast features gain strength when compressed (from planar rotation) and enhance their attraction to the preconditioning edge detector. The conex structure and roads with obvious straight lines and seemingly high contrast were seldom chosen for long as the best candidate for being a line. Hough line detection was dismissed as a viable method of image pair correlation.

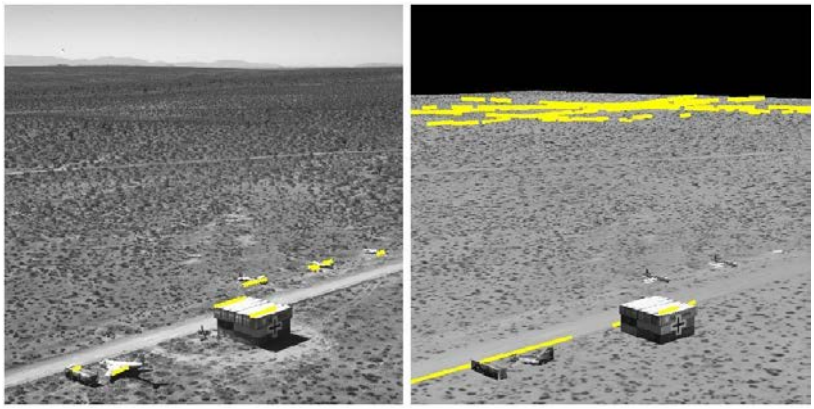
5.3.4 SIFT® Feature Point Matching. SIFT® matching between real and synthetic images demonstrated a high-degree of fidelity in all considered flight test data runs. Figure 5.7 shows three real and synthetic image pairs with matching SIFT® points. While SIFT® pairs in Figures 5.7(a) and (b) may seem sparse, it should be noted that the number of pairs displayed was significantly reduced to be more easily seen here for illustrative purposes; the setting that was actually used filled much of the screen with SIFT® point matches. Figure 5.7 (c) demonstrates how the number of matches increases with decreasing range to the target, provided the target and its environment are "interesting" to the SIFT® algorithm. A balance



(a) Real (left) and synthetic (right) images early in a conex run with Hough lines.



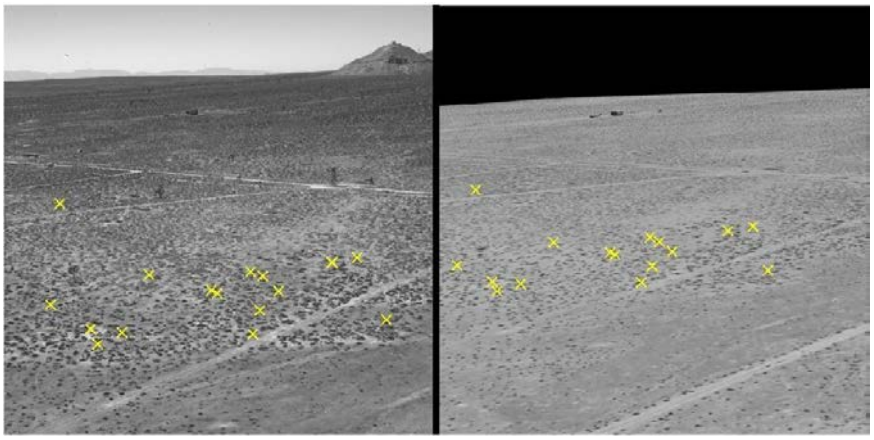
(b) Real (left) and synthetic (right) images midway through a conex run with Hough lines.



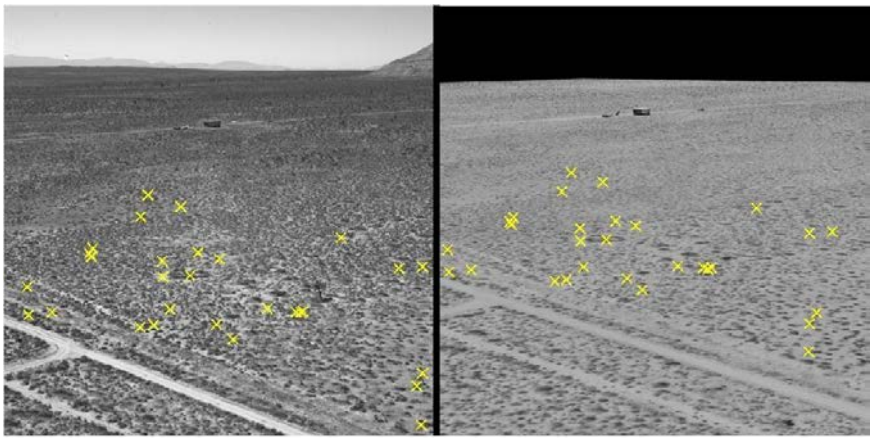
(c) Real (left) and synthetic (right) images late in a conex run with Hough lines.

Figure 5.6: Hough lines in real and synthetic images on a run against the conex target. After several pre-processing steps were performed on the images, a Hough line-detection transform was applied. The synthetic image preferred lines in the distance.

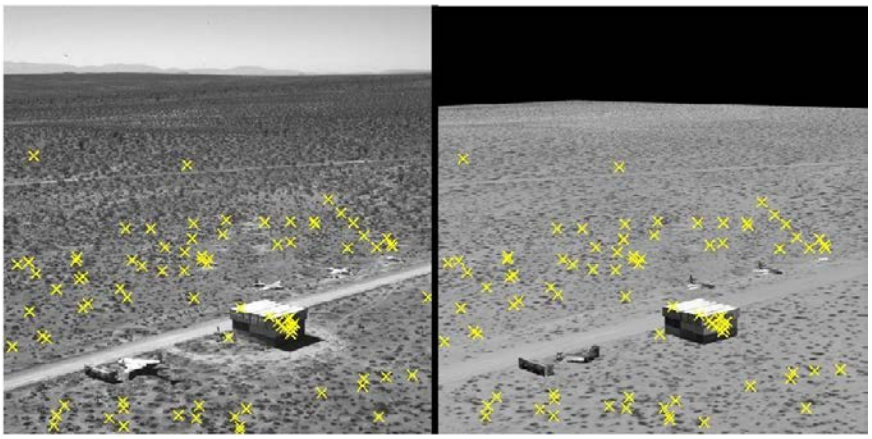
between generating a well-populated SIFT® data set and generating so many pairs that significant errors are introduced was determined experimentally. Unlike experimentally derived results in previous image comparison techniques discussed, however, the settings appeared to be consistently optimal. It should also be noted that only pixel areas of interest to the algorithm matter; even if the model is incomplete, as shown in Figure 5.28.



(a) Real (left) and synthetic (right) images early in a conex run with SIFT® matches.

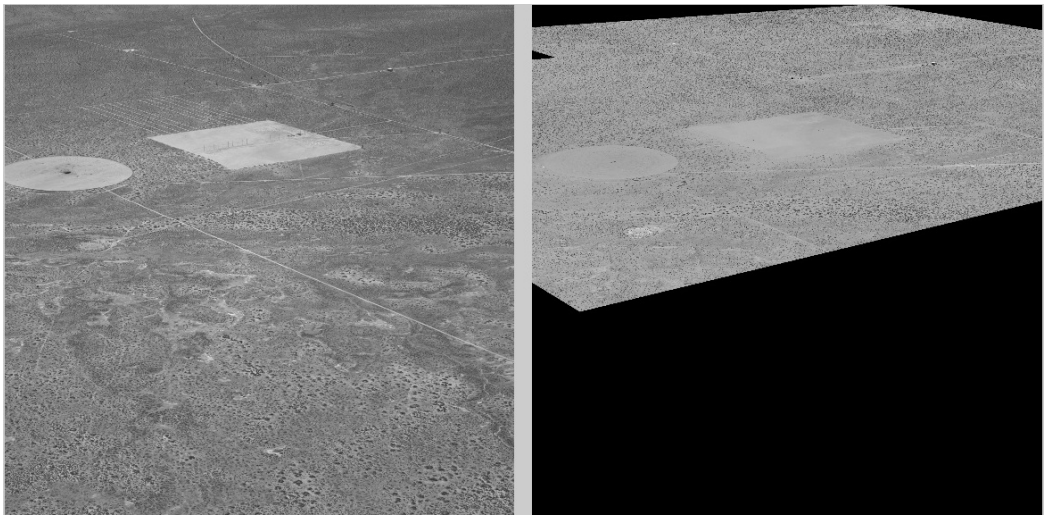


(b) Real (left) and synthetic (right) images midway through a conex run with SIFT® matches.

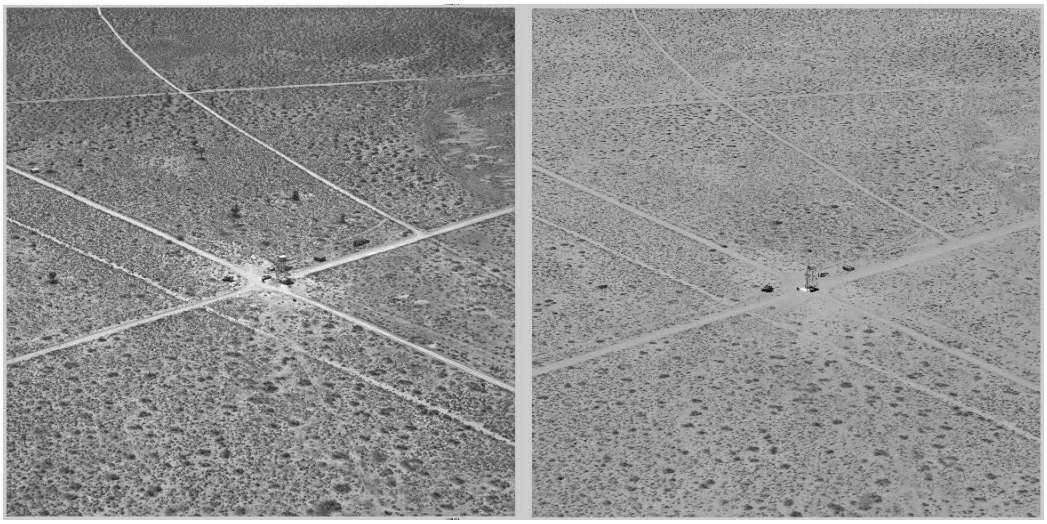


(c) Real (left) and synthetic (right) images late in a conex run with SIFT® matches.

Figure 5.7: SIFT® matched features in real and synthetic images on a run against the conex target. The threshold for matches is turned up for illustration purposes; available correct matches are usually much greater in number.



(a) Cowbell 210° weapon profile at beginning with reduced canvas; real (left), synthetic (right).



(b) Cowbell 210° closing on target with matching and navigation updates provided by the VANSPR algorithm; real (left), synthetic (right).

Figure 5.8: Image matching far and near. The image processing portion of the VANSPR algorithm has no concept of range; it treats the real and synthetic image pairs simply as two similar images. The navigation portion of the algorithm, however, is aware of range and makes image perturbations and measurement weightings accordingly. Note that the algorithm still operates with incomplete information (top).

5.4 Performance of the VANSPR Algorithm

After the rending portion of the algorithm was validated to ensure a reasonable match would be presented given truth data, the UKF-based prediction mechanization was ready for test using only the time-stamped inertial Δv and $\Delta\theta$ measurements. Again, the goal of this testing in the early phases of SPR using *a priori* world models was to simply beat an INS-only solution. While this may sound trivial, the algorithm must be assembled in such a way as to correctly pass true, albeit noisy, measurement updates and confidences (covariances) for optimal weightings.

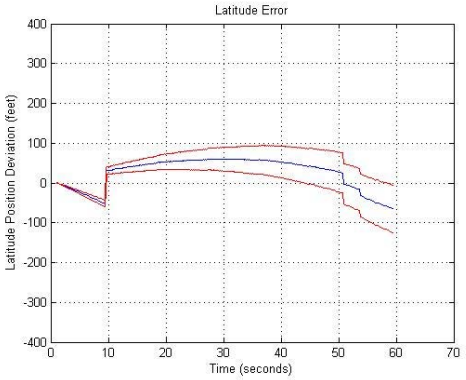
5.4.1 Measurement Scheduling. Most of the runs consisted of about 205 time-stamped images. All of these images were not processed for each run. Instead, a standard schedule was devised that was adhered to on all runs. The reasons that all images were not used include:

1. Pixel shift due to image perturbation is much smaller at larger distances from the target and of a lower discernible resolution. Therefore, the amount of actionable information that can be extracted from multiple image comparisons at large distances is relatively small.
2. Less SIFT® points are available at larger distances from the target which equates to greater measurement uncertainty. The algorithm considers less than 10 measurements as no measurements.
3. A MATLAB® 3D Simulink Animation Toolbox software bug causes periodic synthetic window update failures. Over the course of data processing and reduction, this glitch occurred approximately one out of every 25 runs. When it did occur, the virtual canvas, from which the synthetic image is captured, blanked out or updated to some impossible pose. Inevitably, this glitch always occurred toward the end of run processing, a task which takes about an hour and a half, thereby invalidating all run data processed up to that point and preventing an observation of the endgame navigation solution.

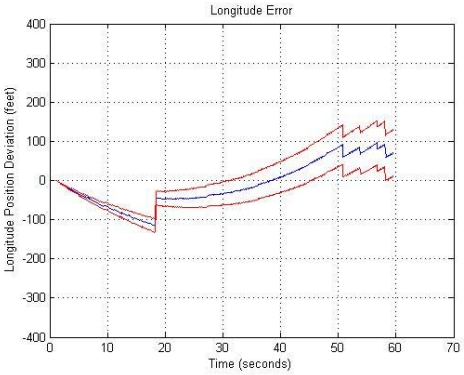
Because of the limited additional value of processing all the measurements early in the run and the additional risk of software failure, the best solution was to create a scheduled measurement process. The measurement schedule time line, discussed in reverse chronological order, was as follows:

1. Three measurements separated by 0.3 seconds before the end of the data collection run.
2. Three measurements separated by 1.5 seconds.
3. Three measurements separated by 3.0 seconds.
4. All preceding measurements separated by 9.0 seconds.

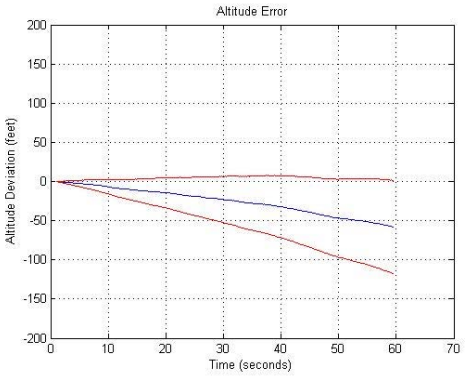
5.4.2 Test Point Results. The full results of the 14 weapon profile flight test runs are shown individually in Figures 5.9-5.22 and corporately in Table 5.2 which shows positional, altitude, and spherical (total) errors for each. The individual figures show latitude error, longitude error, altitude error, spherical error, the number of SIFT® matches found at each measurement update, and an overhead view of the trajectory for reader reference. The jumpiness seen in the data is due to the ratio of $\mathbf{Q} : \mathbf{R}$ and changes depending on filter tuning.



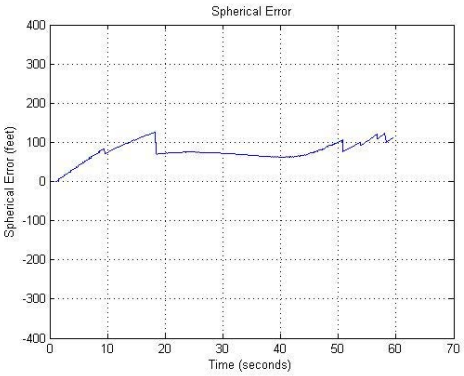
(a) Latitude error with $one - \sigma$ uncertainty.



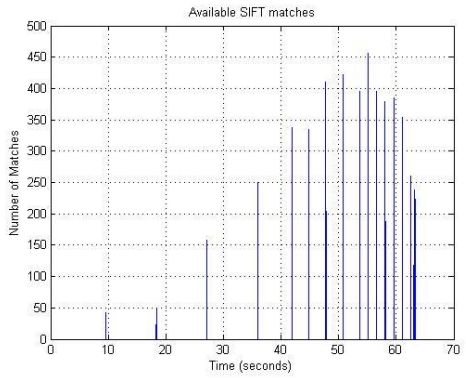
(b) Longitude error with $one - \sigma$ uncertainty.



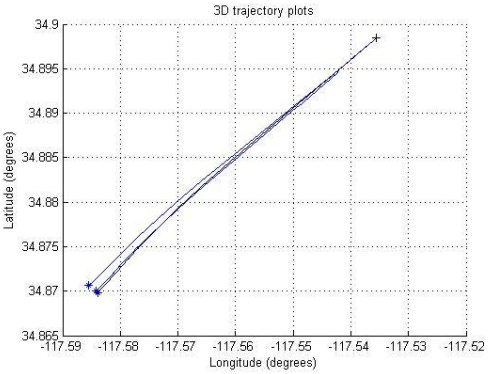
(c) Altitude error with $one - \sigma$ uncertainty.



(d) Spherical error.

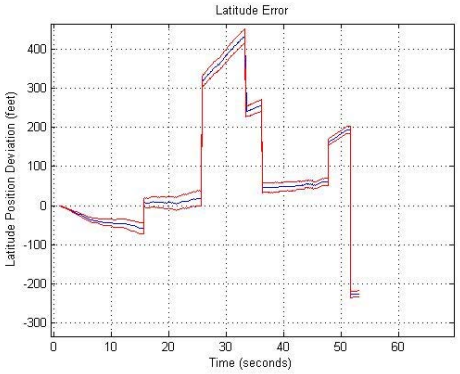


(e) SIFT® matches available.

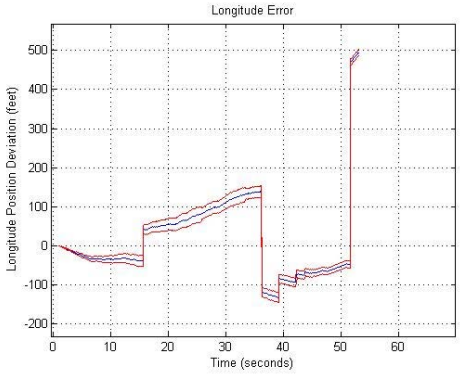


(f) Overhead view comparing TSPI (blue), VANSPR (discontinous black) and INS-only (black).

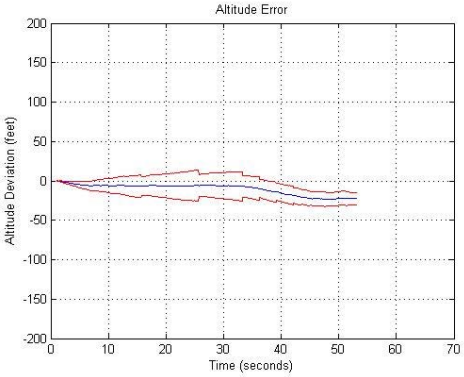
Figure 5.9: SAEC 223° position error, feature matches, and overhead view. Measurement uncertainty was $\sigma_x = \sigma_y = 20m$. The scheduling of measurements and amount of available SIFT measurements have a significant effect on the ability of VANSPR to correct the position error.



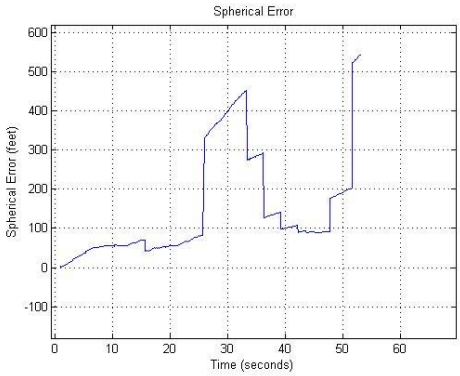
(a) Latitude error with $one-\sigma$ uncertainty.



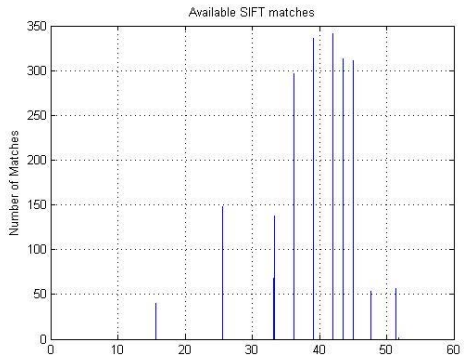
(b) Longitude error with $one-\sigma$ uncertainty.



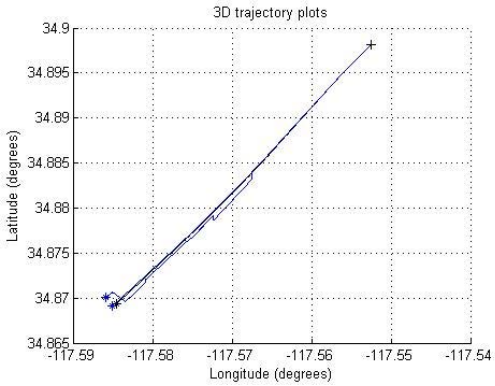
(c) Altitude error with $one-\sigma$ uncertainty.



(d) Spherical error.

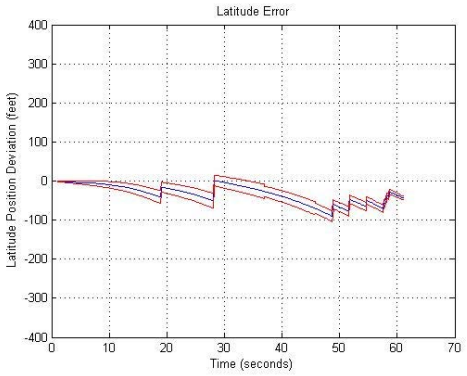


(e) SIFT® matches available.

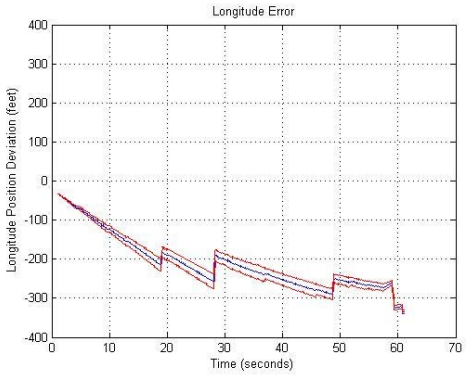


(f) Overhead view comparing TSPI (blue), VANSPR (discontinous black) and INS-only (black)

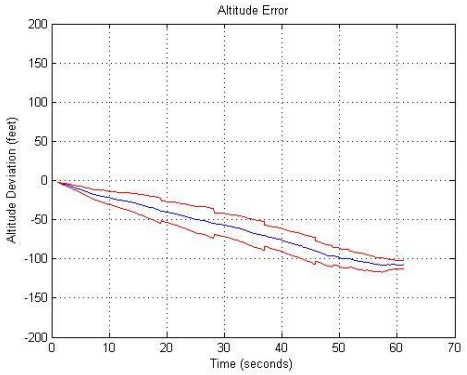
Figure 5.10: SAEC 208° position error, feature matches, and overhead view. Measurement uncertainty is σ_x=20m,σ_y=20m. The scheduling of measurements and amount of available SIFT measurements have a significant effect on the ability of VANSPR to correct the position error.



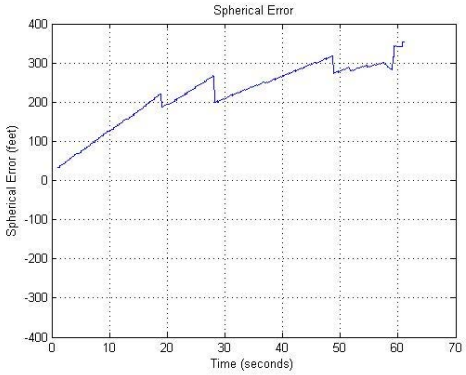
(a) Latitude error with $one - \sigma$ uncertainty.



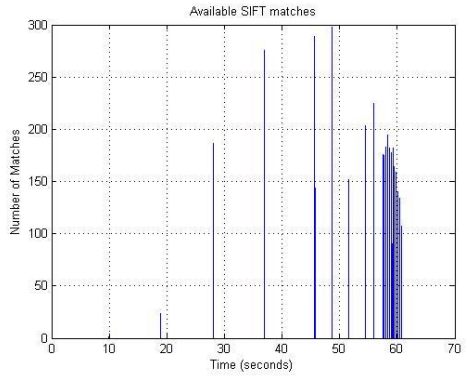
(b) Longitude error with $one - \sigma$ uncertainty.



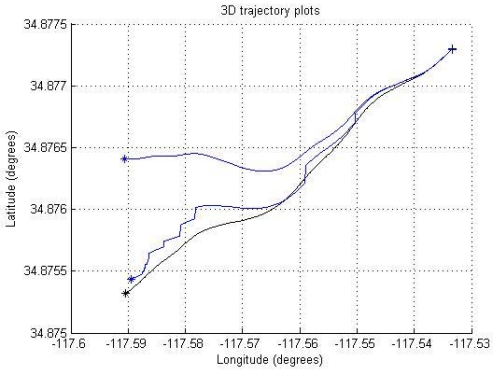
(c) Altitude error with $one - \sigma$ uncertainty.



(d) Spherical error.

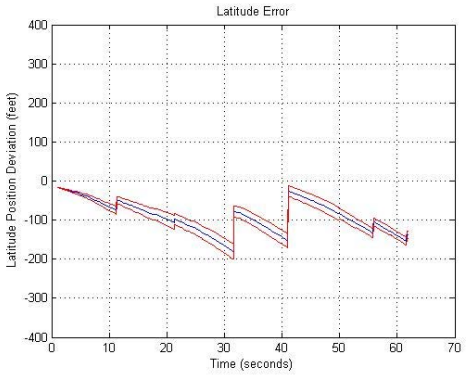


(e) SIFT® matches available.

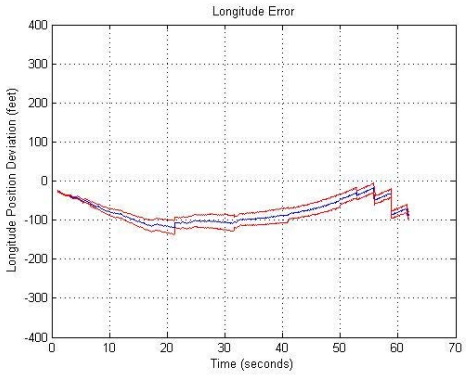


(f) Overhead view comparing TSPI (blue), VANSPR (discontinous black) and INS-only (black)

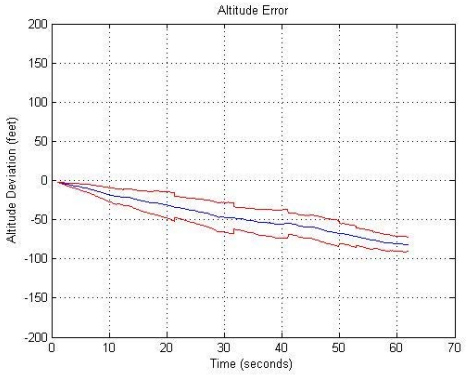
Figure 5.11: Conex 255° position error, feature matches, and overhead view. Measurement uncertainty is $\sigma_x = \sigma_y = 20m$. The scheduling of measurements and amount of available SIFT measurements have a significant effect on the ability of VANSPR to correct the position error.



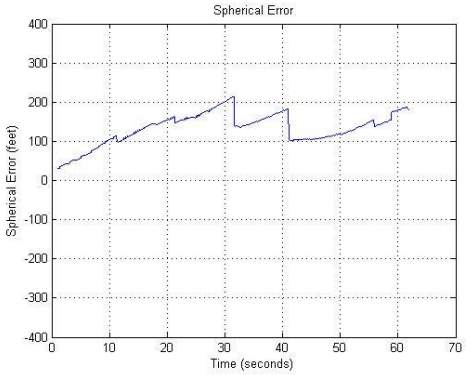
(a) Latitude error with $one - \sigma$ uncertainty.



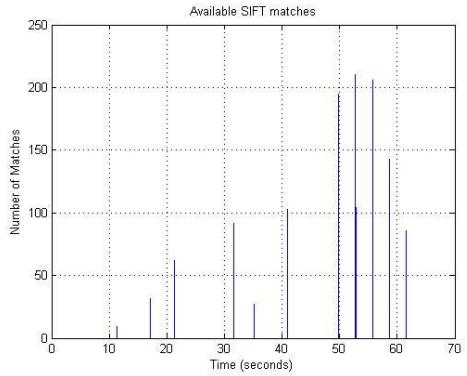
(b) Longitude error with $one - \sigma$ uncertainty.



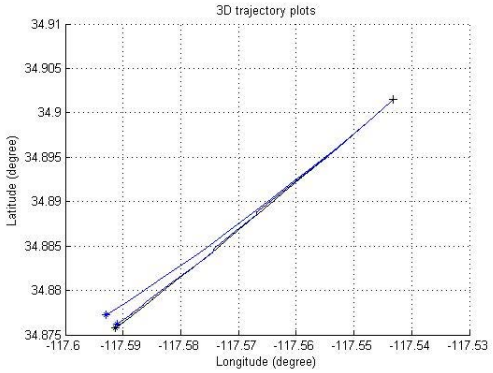
(c) Altitude error with $one - \sigma$ uncertainty.



(d) Spherical error.

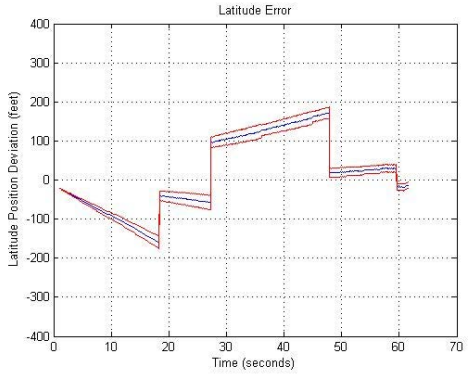


(e) SIFT® matches available.

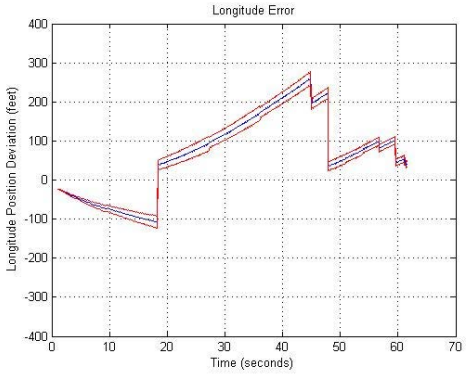


(f) Overhead view comparing TSPI (blue), VANSPR (discontinous black) and INS-only (black)

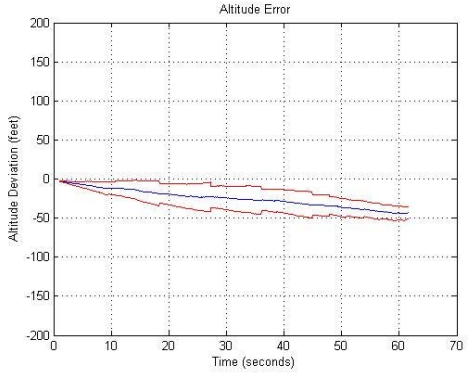
Figure 5.12: Conex 225° position error, feature matches, and overhead view. Measurement uncertainty is $\sigma_x = \sigma_y = 20m$. The scheduling of measurements and amount of available SIFT measurements have a significant effect on the ability of VANSPR to correct the position error.



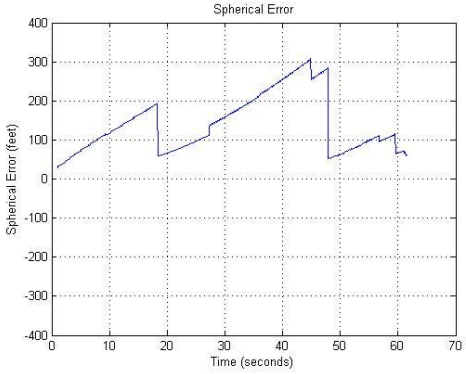
(a) Latitude error with $one - \sigma$ uncertainty.



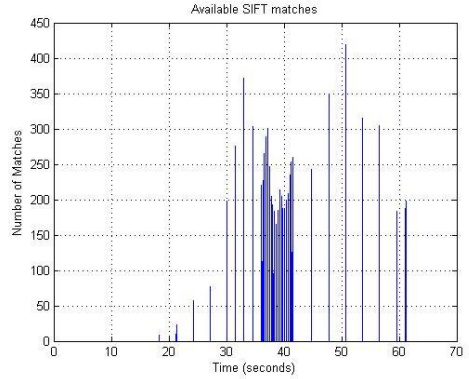
(b) Longitude error with $one - \sigma$ uncertainty.



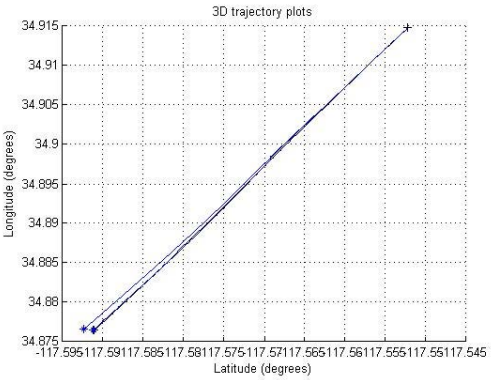
(c) Altitude error with $one - \sigma$ uncertainty.



(d) Spherical error.

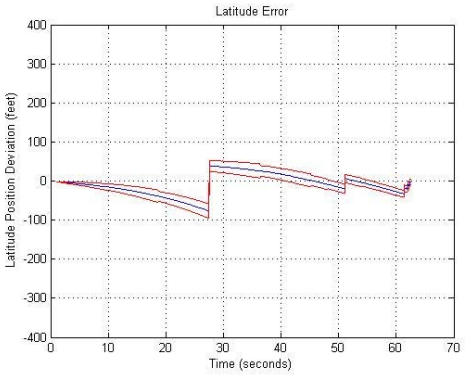


(e) SIFT® matches available.

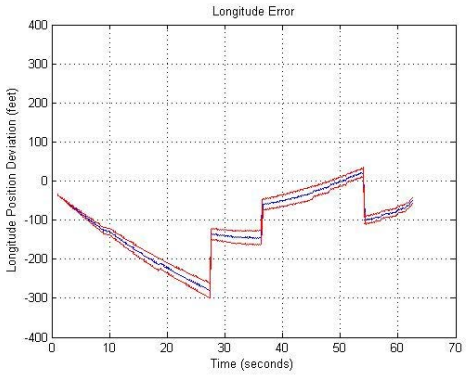


(f) Overhead view comparing TSPI (blue), VANSPR (discontinous black) and INS-only (black)

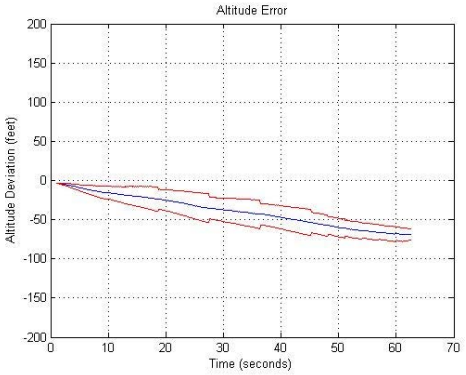
Figure 5.13: Conex 210° position error, feature matches, and overhead view. Measurement uncertainty is $\sigma_x = \sigma_y = 20m$. The scheduling of measurements and amount of available SIFT measurements have a significant effect on the ability of VANSPR to correct the position error.



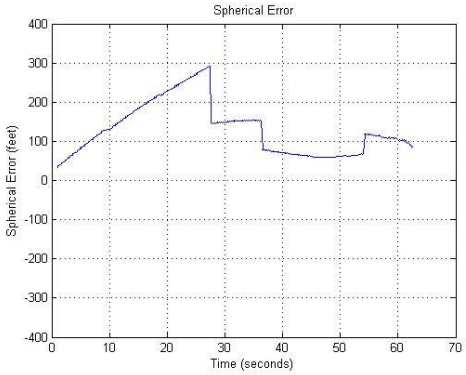
(a) Latitude error with $one - \sigma$ uncertainty.



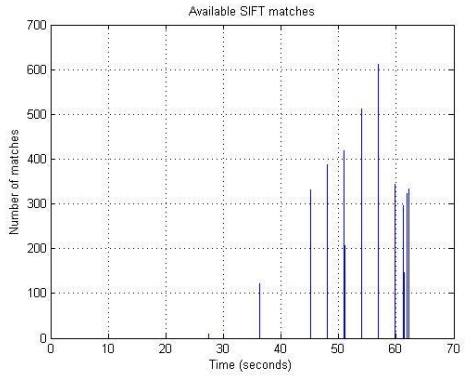
(b) Longitude error with $one - \sigma$ uncertainty.



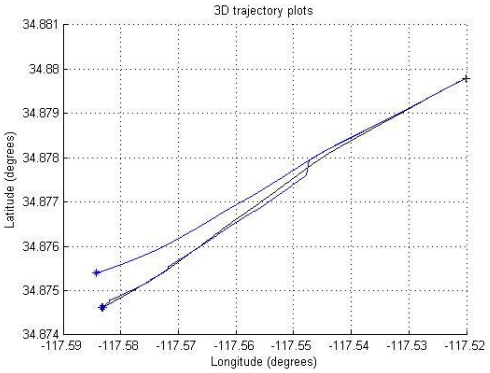
(c) Altitude error with $one - \sigma$ uncertainty.



(d) Spherical error.

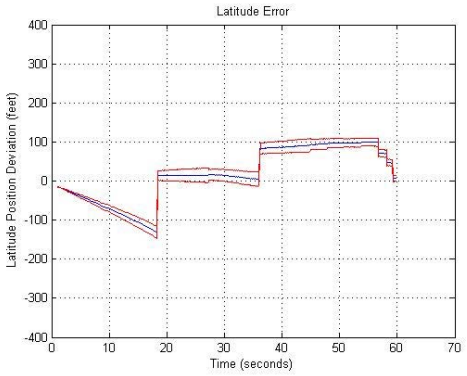


(e) SIFT® matches available.

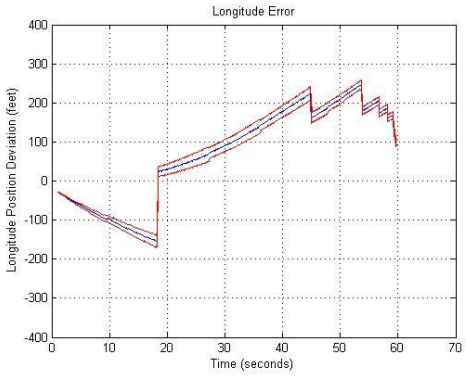


(f) Overhead view comparing TSPI (blue), VANSPR (discontinous black) and INS-only (black)

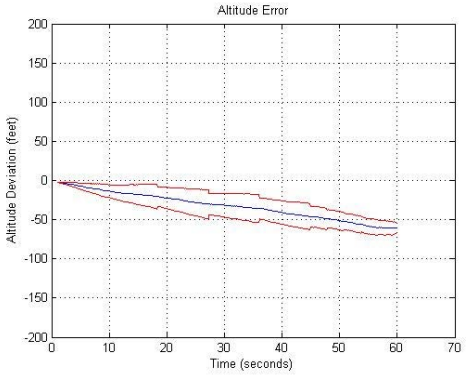
Figure 5.14: Cowbell Tower 255° position error, feature matches, and overhead view. Measurement uncertainty is $\sigma_x = \sigma_y = 20m$. The scheduling of measurements and amount of available SIFT measurements have a significant effect on the ability of VANSPR to correct the position error.



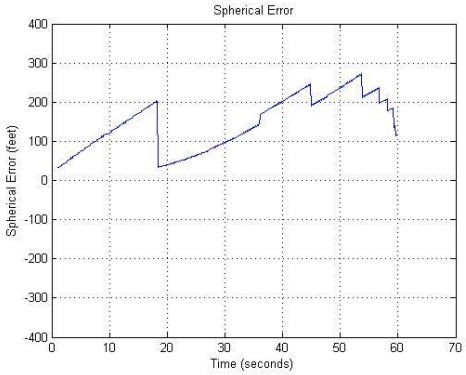
(a) Latitude error with $one - \sigma$ uncertainty.



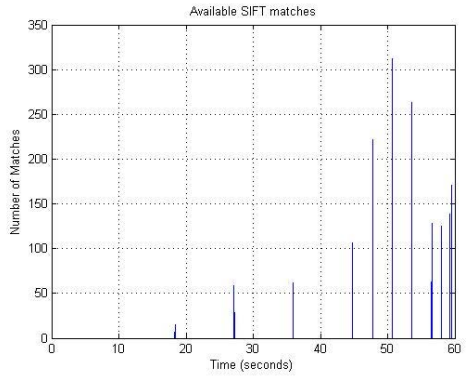
(b) Longitude error with $one - \sigma$ uncertainty.



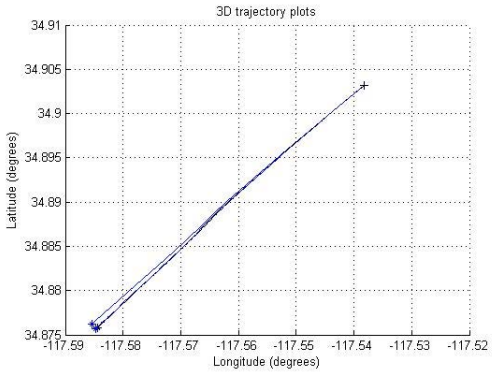
(c) Altitude error with $one - \sigma$ uncertainty.



(d) Spherical error.

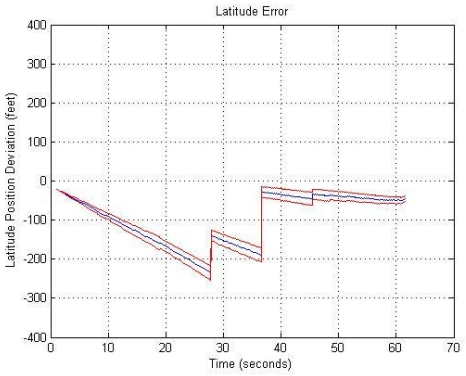


(e) SIFT® matches available.

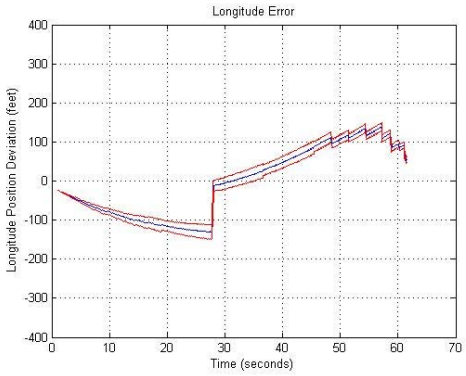


(f) Overhead view comparing TSPI (blue), VANSPR (discontinous black) and INS-only (black)

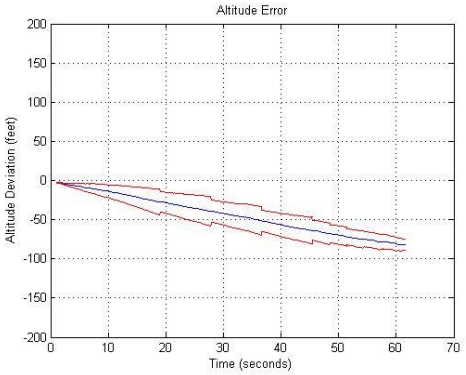
Figure 5.15: Cowbell Tower 225° position error, feature matches, and overhead view. Measurement uncertainty is $\sigma_x = \sigma_y = 20m$. The scheduling of measurements and amount of available SIFT measurements have a significant effect on the ability of VANSPR to correct the position error.



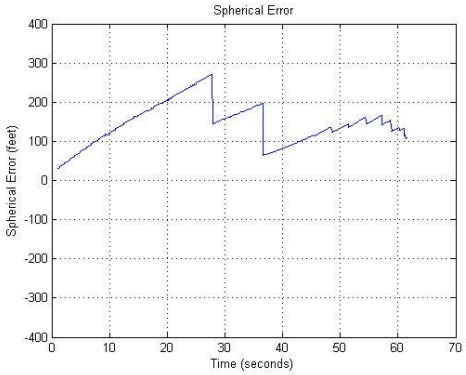
(a) Latitude error with $one - \sigma$ uncertainty.



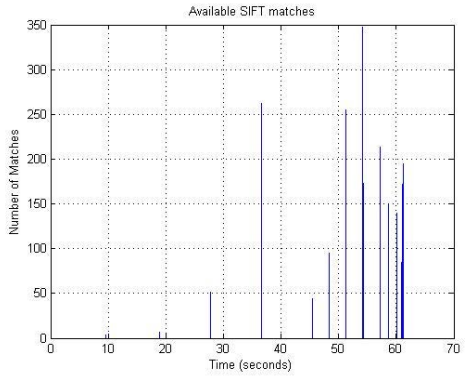
(b) Longitude error with $one - \sigma$ uncertainty.



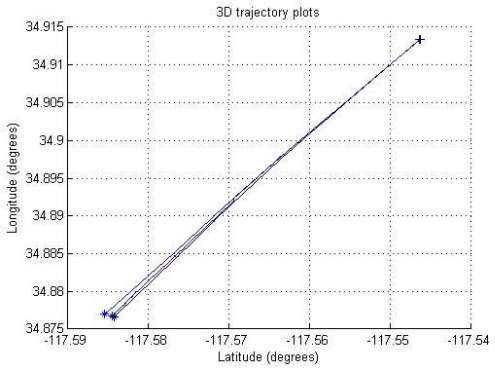
(c) Altitude error with $one - \sigma$ uncertainty.



(d) Spherical error.

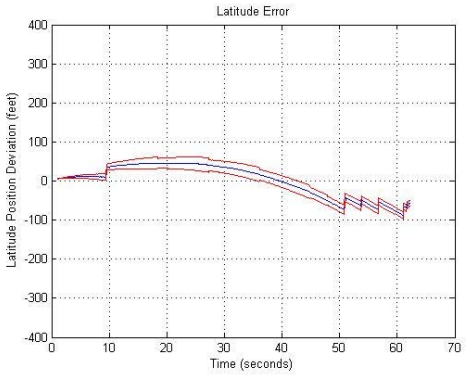


(e) SIFT® matches available.

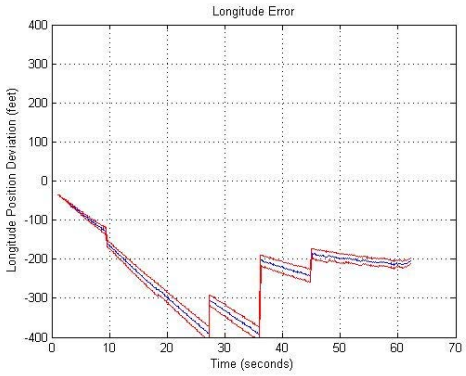


(f) Overhead view comparing TSPI (blue), VANSPR (discontinous black) and INS-only (black)

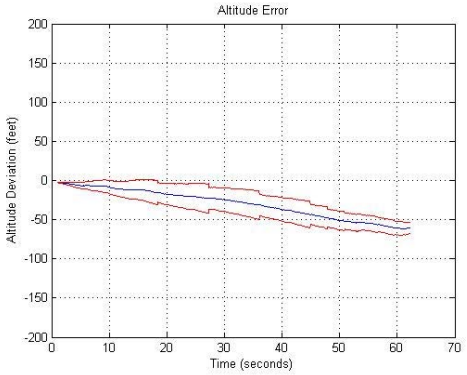
Figure 5.16: Cowbell Tower 210° position error, feature matches, and overhead view. Measurement uncertainty is $\sigma_x = \sigma_y = 20m$. The scheduling of measurements and amount of available SIFT measurements have a significant effect on the ability of VANSPR to correct the position error.



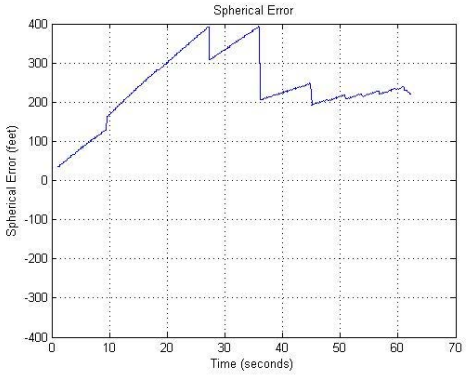
(a) Latitude error with $one - \sigma$ uncertainty.



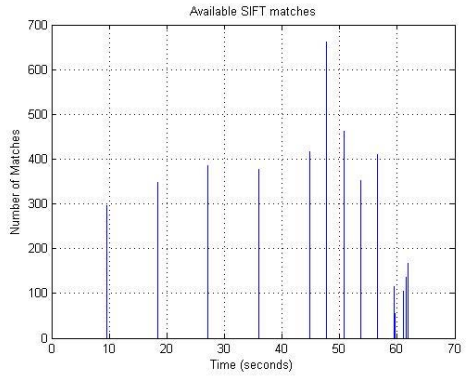
(b) Longitude error with $one - \sigma$ uncertainty.



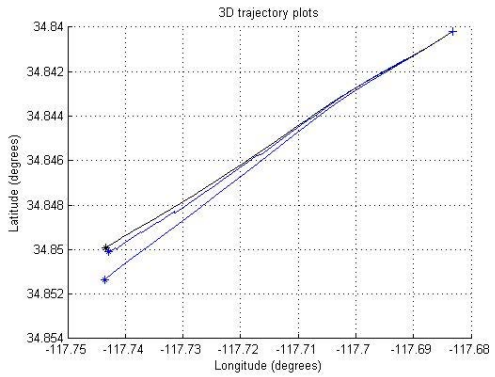
(c) Altitude error with $one - \sigma$ uncertainty.



(d) Spherical error.

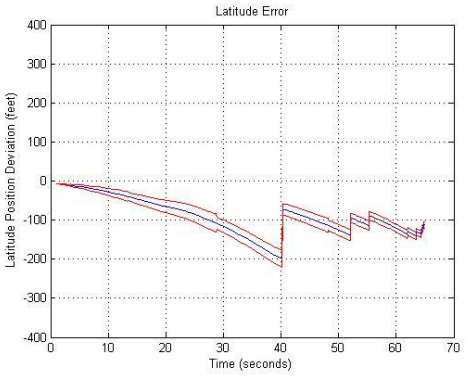


(e) SIFT® matches available.

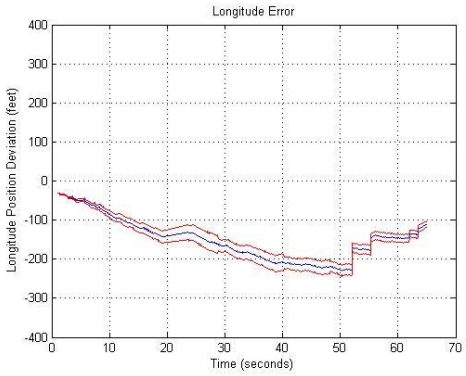


(f) Overhead view comparing TSPI (blue), VANSPR (discontinous black) and INS-only (black)

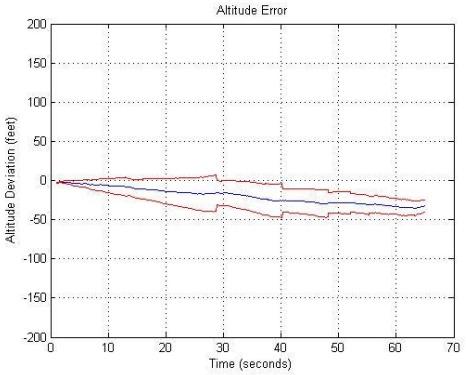
Figure 5.17: Tank 270° position error, feature matches, and overhead view. Measurement uncertainty is $\sigma_x = \sigma_y = 20m$. The scheduling of measurements and amount of available SIFT measurements have a significant effect on the ability of VANSPR to correct the position error.



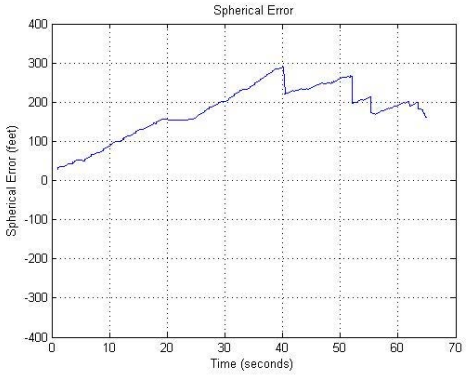
(a) Latitude error with $one - \sigma$ uncertainty.



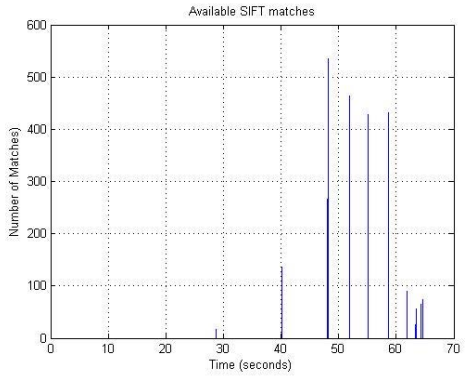
(b) Longitude error with $one - \sigma$ uncertainty.



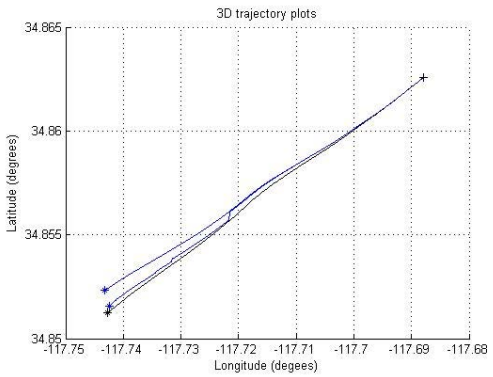
(c) Altitude error with $one - \sigma$ uncertainty.



(d) Spherical error.

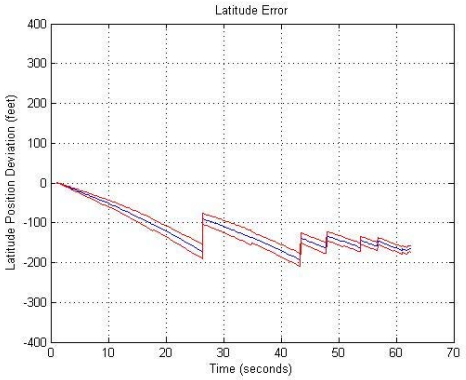


(e) SIFT® matches available.

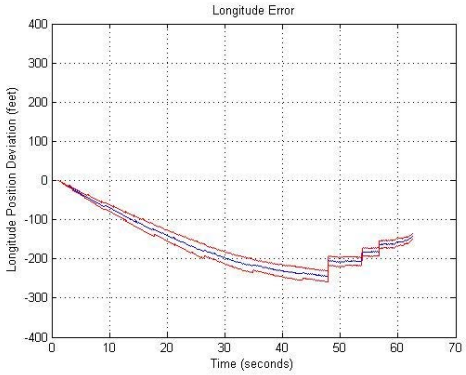


(f) Overhead view comparing TSPI (blue), VANSPR (discontinous black) and INS-only (black)

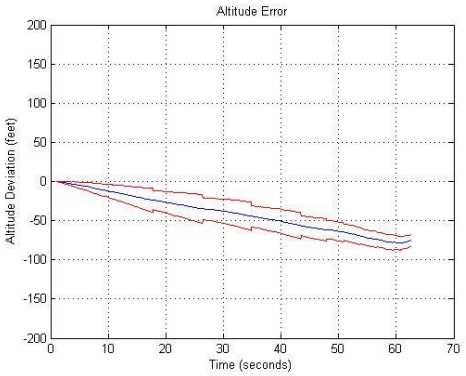
Figure 5.18: Tank 240° position error, feature matches, and overhead view. Measurement uncertainty is $\sigma_x = \sigma_y = 20m$. The scheduling of measurements and amount of available SIFT measurements have a significant effect on the ability of VANSPR to correct the position error.



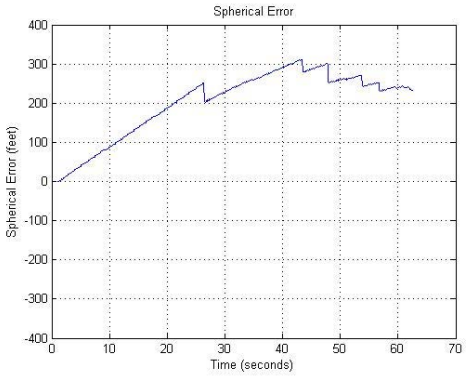
(a) Latitude error with $one - \sigma$ uncertainty.



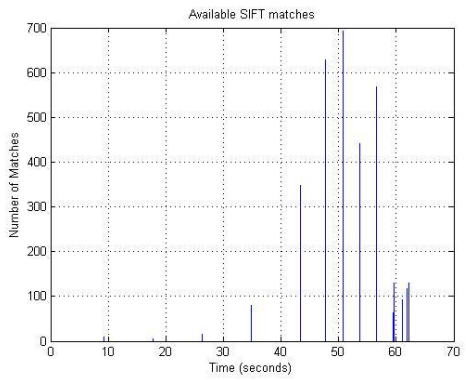
(b) Longitude error with $one - \sigma$ uncertainty.



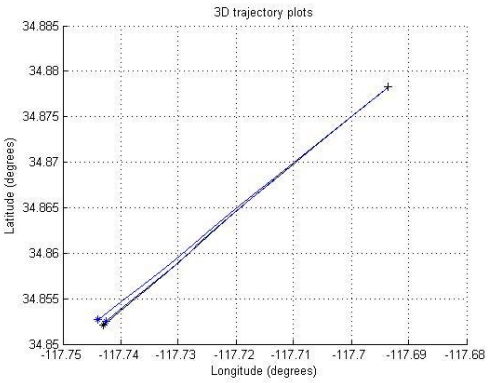
(c) Altitude error with $one - \sigma$ uncertainty.



(d) Spherical error.

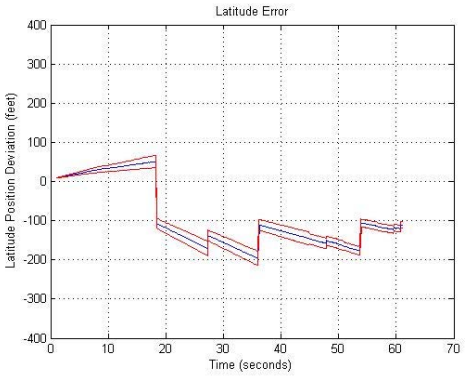


(e) SIFT® matches available.

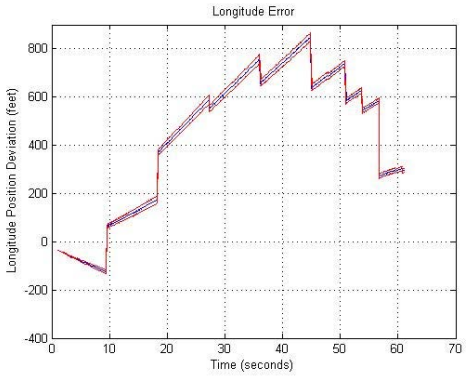


(f) Overhead view comparing TSPI (blue), VANSPR (discontinous black) and INS-only (black)

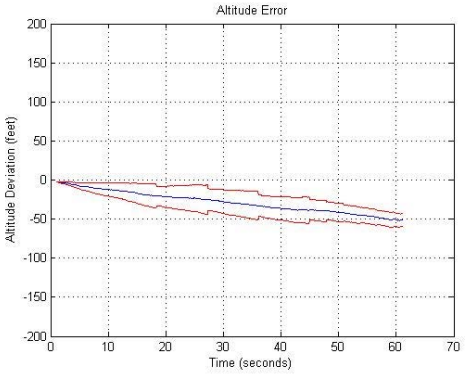
Figure 5.19: Tank 225° position error, feature matches, and overhead view. Measurement uncertainty is $\sigma_x = \sigma_y = 20m$. The scheduling of measurements and amount of available SIFT measurements have a significant effect on the ability of VANSPR to correct the position error.



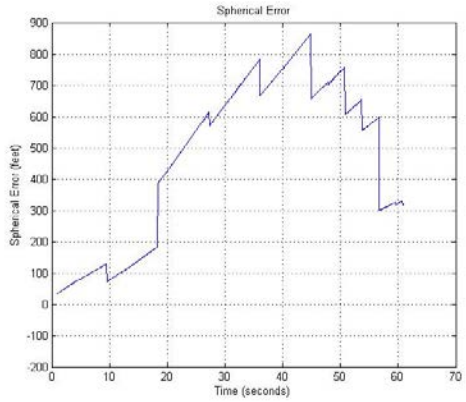
(a) Latitude error with $one - \sigma$ uncertainty.



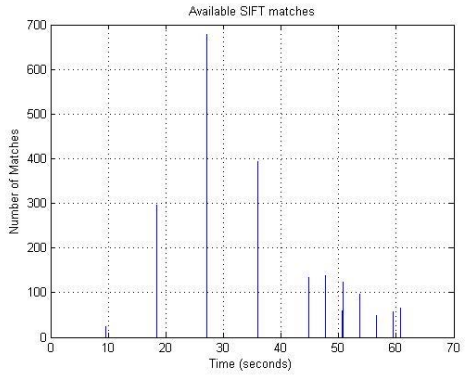
(b) Longitude error with $one - \sigma$ uncertainty.



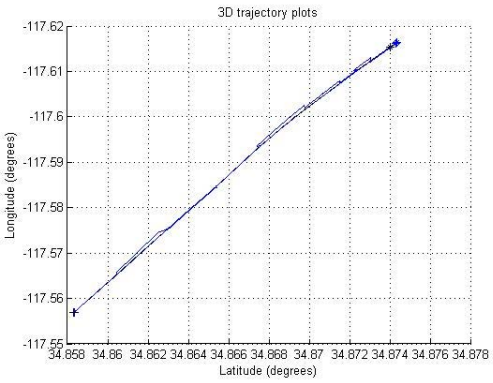
(c) Altitude error with $one - \sigma$ uncertainty.



(d) Spherical error.

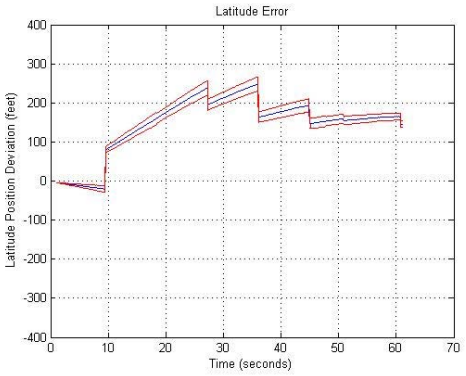


(e) SIFT® matches available.

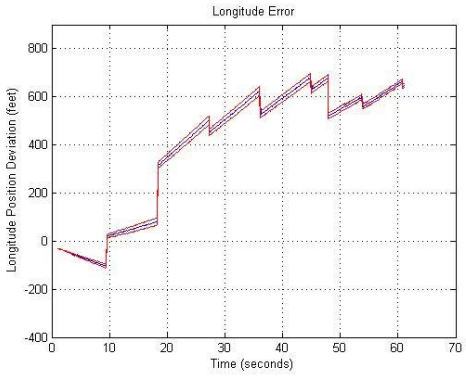


(f) Overhead view comparing TSPI (blue), VANSPR (discontinous black) and INS-only (black)

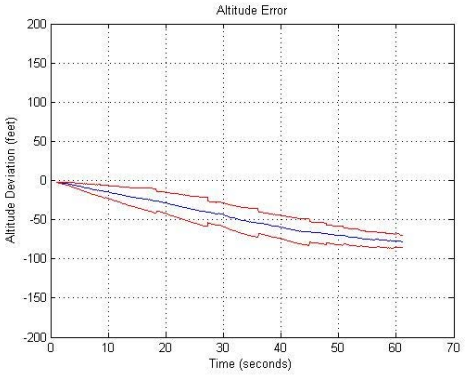
Figure 5.20: X-33 275° position error, feature matches, and overhead view. Measurement uncertainty is $\sigma_x = \sigma_y = 20m$. The scheduling of measurements and amount of available SIFT measurements have a significant effect on the ability of VANSPR to correct the position error.



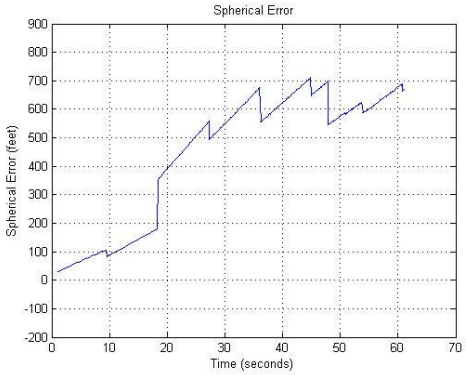
(a) Latitude error with $one - \sigma$ uncertainty.



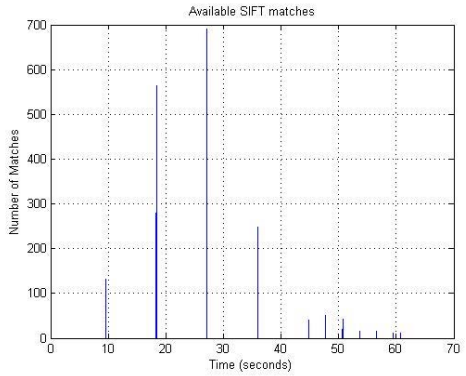
(b) Longitude error with $one - \sigma$ uncertainty.



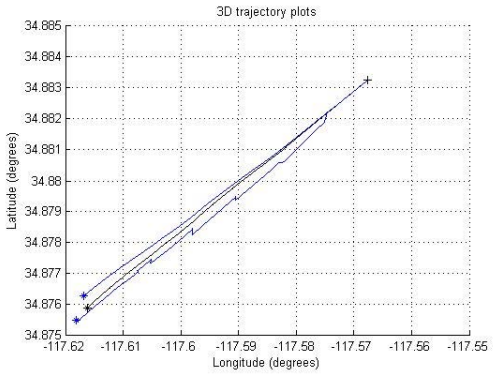
(c) Altitude error with $one - \sigma$ uncertainty.



(d) Spherical error.

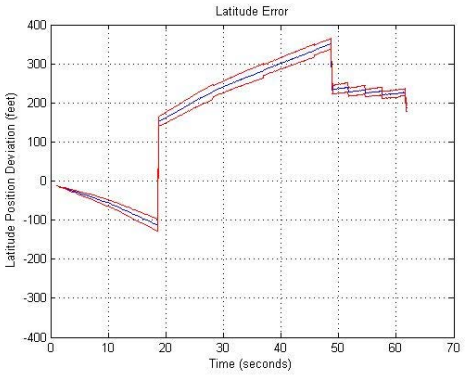


(e) SIFT® matches available.

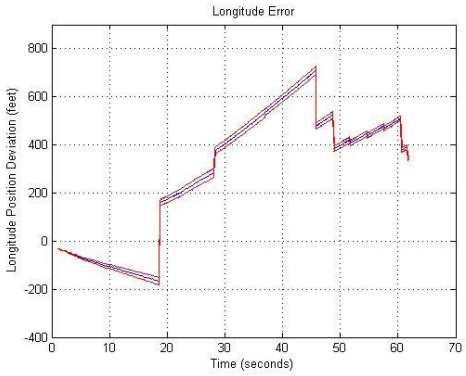


(f) Overhead view comparing TSPI (blue), VANSPR (discontinous black) and INS-only (black)

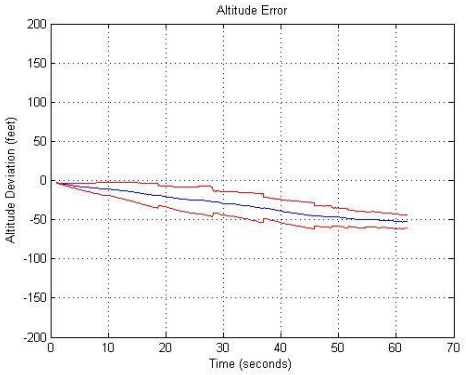
Figure 5.21: X-33 245° position error, feature matches, and overhead view. Measurement uncertainty is $\sigma_x = \sigma_y = 20m$. The scheduling of measurements and amount of available SIFT measurements have a significant effect on the ability of VANSPR to correct the position error.



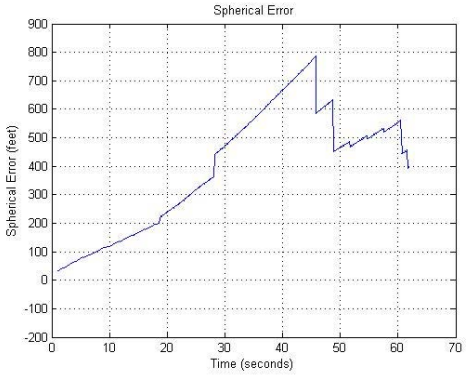
(a) Latitude error with $one - \sigma$ uncertainty.



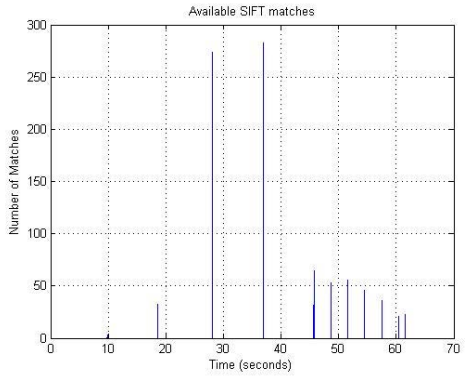
(b) Longitude error with $one - \sigma$ uncertainty.



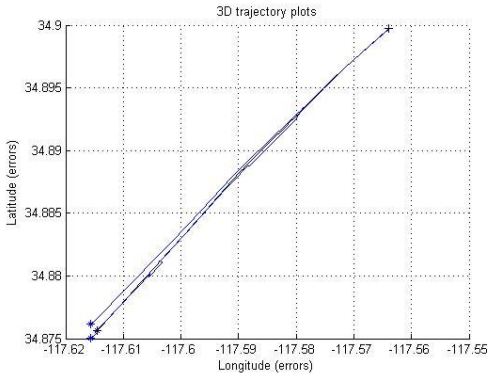
(c) Altitude error with $one - \sigma$ uncertainty.



(d) Spherical error.



(e) SIFT® matches available.



(f) Overhead view comparing TSPI (blue), VANSPR (discontinous black) and INS-only (black)

Figure 5.22: X-33 230° position error, feature matches, and overhead view. Measurement uncertainty is $\sigma_x = \sigma_y = 20m$. The scheduling of measurements and amount of available SIFT measurements have a significant effect on the ability of VANSPR to correct the position error.

Table 5.2: Full VANSPR results. The errors shown are position, altitude, and spherical at the end of runs. Measurement error was standardized at 20 m for best general use; better results for specific runs were obtained using other values and measurement uncertainty determination methods.

Target	INS error (ft)	VANSPR error (ft)
SAEC 223°	563, 42, 564	101, -59, 118
SAEC 208°	182, 22, 183	470, -22, 470
Conex 255°	727, 1, 727	141, -79, 161
Conex 225°	405, -37, 407	262, -105, 282
Conex 210°	392, 38, 393	49, -40, 63
Cowbell 255°	412, 34, 414	23, -65, 70
Cowbell 225°	543, 13, 543	101, -57, 116
Cowbell 210°	378, 3, 378	66, -78, 103
Tank 270°	530, 50, 532	164, -57, 174
Tank 240°	431, 7, 431	135, -30, 138
Tank 225°	372, 16, 372	190, -74, 204
X-33 275°	245, 32, 247	293, -48, 297
X-33 245°	230, 2, 230	575, -76, 581
X-33 230°	370, 27, 370	364, -50, 368

5.4.2.1 General Observations. Several general observations can be made on the data from the individual runs that may provide insight into further analysis. First, a comment on accuracy must be made. In the case of seemingly bad result of 470 feet for the SAEC 208° run, the fact that the almost identical runs of Conex 210° and Cowbell Tower 210° both had their best results of 63 feet and 103 feet, respectively, seems suspicious. Further investigation was not conducted in the interest of time for this research but should be settled for future work.

A strong correlation can be seen on most of the runs between a high number of available SIFT points and low error. Where the number of SIFT points peak, the error curves usually trend back towards zero. The X-33 runs are the clearest example of the effect a small pool of available SIFT matches has on the ability of VANSPR to correct itself. An additional comment on the X-33 runs is that there was a considerable difference in the appearance of the complex due to the changes that were made between the time the geo-orthorectified imagery tile photographs were taken in 2008 and when the test flights were accomplished in September 2010.

Mission planning for the targets was focused on the aspect to a selected target surface and not to cardinal heading. All the runs were conducted from magnetic headings of 208° to 275°, so observability into some of the directional error effects is limited. Most of the runs have a southerly error that seems to worsen as the flight direction becomes more westerly. The longitudinal errors were considerably higher in general, revealing a "long-short" (actually long for all cases seen here). The side-to-side errors were generally well-contained and corrected for.

Note that although the total error is lower using VANSPR for all but two cases, the altitude errors are generally higher. This is probably due to not taking an altitude measurement directly. It has already been explained that observational errors were the primary factor for this simplification. Depending on the use of an air-to-ground weapon that could make use of VANSPR, altitude error is of variable importance. If a 90° impact angle is desired and fusing is not dependent on self-contained timing,

positional error is most important. For applications requiring limited maneuvering in mountainous or urban terrain, altitude errors can be the difference between bombs on target and striking an obstacle well short of the intended target. Table 5.3 refers to the same runs but focuses on total (spherical) error.

Table 5.3: Spherical error results using the same parameters. Bad data is indicated by "XX". Position and altitude errors are shown separately in Table 5.2. $0°, 30°$, and $45°$ refer to run-in headings as described in Chapter 4.

	Weapon Profile Final Error (feet)					
Target	0°		30°		45°	
Method	INS	VANSPR	INS	VANSPR	INS	VANSPR
SAEC	XX	XX	564	118	183	470
Conex	727	161	407	282	393	63
Cowbell Tower	414	70	543	116	378	103
Tank	532	174	431	138	372	204
X-33 compound	247	297	230	581	370	368

It may also be of interest to discuss the results in terms of percentage improvement. Table 5.4 shows the same data as Table 5.3, but in terms of percentage improvement, defined as

$$\%_{improvement} = \frac{VANSPR - INS}{INS} \times 100 \tag{5.1}$$

Table 5.4: Spherical error results in percentage improvement. Bad data is indicated by "XX". Measurement uncertainty was $\sigma_x = \sigma_y = 20m$. $0°, 30°$, and $45°$ refer to run-in headings

	Percentage Improvement		
Target	0°	30°	45°
SAEC	XX	79	-156
Conex	78	44	84
Cowbell Tower	83	79	73
Tank	67	68	45
X-33 compound	-20	-153	1

While the data set is not large, analysis by correlation may at least bring out some factors which contributed to the algorithm's success or failure. The average overall VANSPR error, using $\sigma_x = \sigma_y = 20m$ was 288 feet as compared to the INS-only error of 411 feet, for an improvement of 30%. If the user is allowed to incorporate the empirical **R** from Chapter 3 for the worst SAEC and X-33 runs, results improve to 166 feet for a 63% overall improvement.

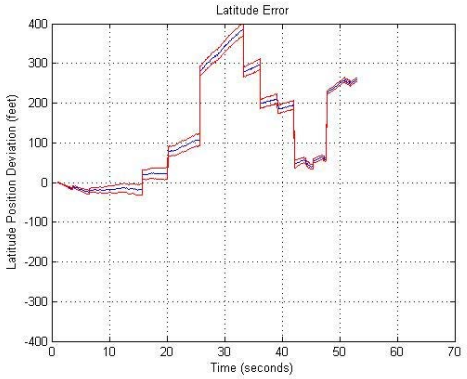
5.4.2.2 Effect of Measurement Scheduling. The measurement scheduling was increased for this run to observe the effects that more measurements would provide. The results are shown in Figure 5.23, for comparison with the nominal measurement schedule for the same target run shown in Figure 5.10. While producing improved results (303 feet versus 470 feet of error), the similar shape indicates problems with measurements at similar times during the run and a more serious problem with the weighting of the measurement itself.

5.4.2.3 Effect of Target Type. A consolidation of INS error, VANSPR error, and percentage improvement dependent on target only is shown in Table 5.5.

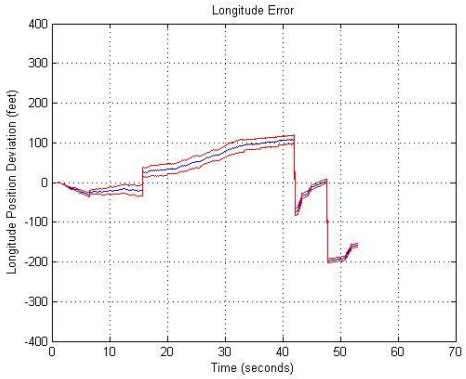
Table 5.5: Percentage error by target. Cowbell Tower was most improved by VANSPR while the X-33 and SAEC targets showed only minimal improvement. Measurement uncertainty was $\sigma_x = \sigma_y = 20m$.

Target	INS error (ft)	VANSPR error (ft)	Percentage Improvement
SAEC	374	588	-57
Conex	509	169	67
Cowbell Tower	445	96	78
Tank	445	172	61
X-33 compound	282	415	-47

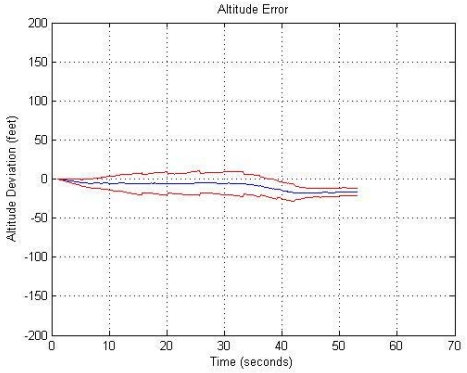
It can be seen that runs against the Cowbell Tower target enjoyed the greatest improvement from VANSPR while the X-33 compound and SAEC board received the least. However, alternate filter tunings give the SAEC target the best improvement. As stated earlier, nearby runs on almost the same heading, such as the Cowbell 210°



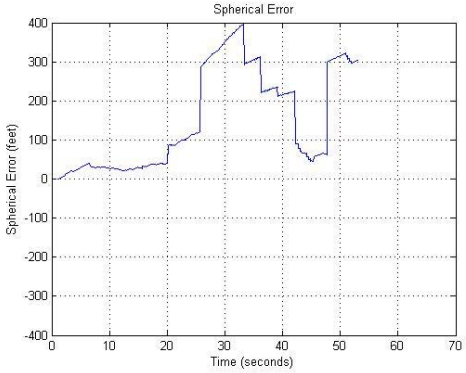
(a) Latitude error with $one - \sigma$ uncertainty.



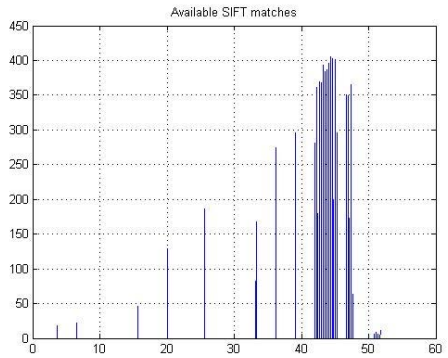
(b) Longitude error with $one - \sigma$ uncertainty.



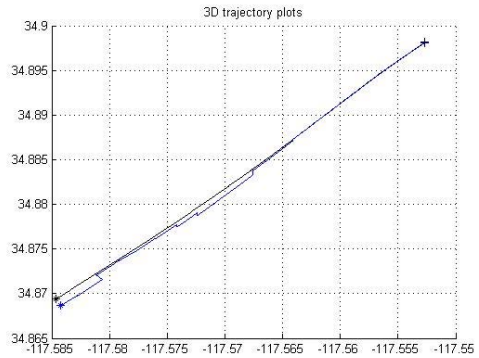
(c) Altitude error with $one - \sigma$ uncertainty.



(d) Spherical error.



(e) SIFT® matches available.



(f) Overhead view comparing TSPI (blue), VANSPR (discontinous black) and INS-only (black)

Figure 5.23: SAEC 208° position error, feature matches, and overhead view with increased measurement scheduling. Measurement uncertainty is $\sigma_x = \sigma_y = 20m$. The measurement scheduling was increased for this run to observe the effects. While producing improved results (303 feet versus 470 feet of error), the similar shape indicates problems with measurements at similar times during the run and a more serious problem with the weighting of the measurement itself.

run, produced a 103 foot error result. Although "targets" were the initial focus because of the weapon guidance context, observation of SIFT® points on runs revealed the great importance of the surrounding terrain - in fact to a greater extent than the targets themselves, especially for the smaller targets. It may seem that the desert floor would be considered as noise or clutter to a visual algorithm, but is in fact rich in visual information, provided the patterns do not change much between current observation and the texture-mapping image tiles used to model the ground.

The effect of chosen measurement noise also contributed to X-33 run results. Using the empirical $\mathbf{R}$, the X-33 endgame errors were 206 feet, 301 feet, and 352 feet on the three aspect runs. The choice of $\sigma_x = \sigma_y = 20m$, which seemed to optimize results from the other targets made the errors worse.

The extremely small contribution of the three-dimensional model on the X-33 run may seem the most surprising. However, this is partially explainable by observing the collected SIFT® points on the target run-in. The SIFT® algorithm chooses a much higher density of points on the desert floor than on the buildings in the X-33 complex.

5.4.3 Effect of Three-dimensional Models. A further examination of the X-33 target was conducted by conducting runs with and without the three-dimensional models present, using the empirical $\mathbf{R}$ matrix which worked better on the X -33 target. The "flat" version was simply the geo-orthorectified imagery tiles texture-mapped onto flat ground panels.

The same run was performed for the 275° X-33 run with a flat environment and with a three-dimensional environment as shown in Figures 5.24 and 5.25 respectively. The results, shown in Table 5.6, indicate almost no difference between the two runs. It should be noted that several of the target environments had changed somewhat from how they appeared in the modeling imagery. Some had temporary vehicles present during flight test data collection or new desert trails forming. The X-33 compound, however, showed the most change; storage containers had been moved and piles of

equipment had been shifted. While many SIFT® points may have been found in the individual real and synthetic images, matched pairs were not abundant.

Table 5.6: Flat versus three-dimensional model. The empirical **R** matrix was used.

Target	Spherical Error (ft)
X-33 three-dimensional	206
X-33 flat	207
X-33 INS only	247

It should also be noted that the target environment that led to the best VANSPR performance with the empirical **R**, the SAEC, is the only completely flat target. The high-contrast SAEC board itself, however, did not yield many SIFT® matches. Although high contrast and distinctive in nature because of it's right angles and symmetry, it would be a better target for pixel-based comparisons. In fact, during experimentation with edge detection on flight test data, the SAEC board was one of the few things recognizable in a binary image.

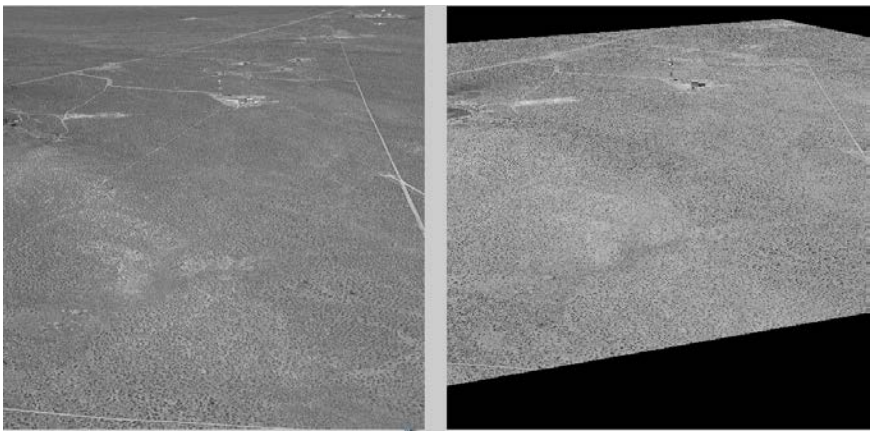
Besides the obvious man-made objects in the environment that would lend themselves to modeling, some natural items, such as trees and mountains, should be considered for modeling. It would likely be a much more difficult task, but could just provide the needed relative positioning required for a precision strike. Figure 5.26 provides an example were visual cues from trees in the environment are missing because they are considered flat. If trees ever were modeled, there would have to be an automatic means to be practical.

5.4.4 Effect of Overexposure. The aperture on the Prosilica 4900 camera was set to full closed on all but the first sortie where it was inadvertently left open one setting value. The resultant overexposure of the images created a unique opportunity to examine the effects. While it was obvious that information was lost on many of the runs, with grayscale images appearing almost binary (only black and white), the tank target seemed to be easier to discern. Therefore, an experiment was conducted

(a) Midway through run on flat X-33 compound; real (left), synthetic (right).

(b) End of run on flat X-33 compound; real (left), synthetic (right).

Figure 5.24: X-33 target run without three-dimensional models. The image is just the geo-orthorectified imagery tiles on a flat ground plane.



(a) Start of run on three-dimensional X-33 compound; real (left), synthetic (right).

(b) Midway through run on three-dimensional X-33 compound; real (left), synthetic (right).

(c) End of run on three-dimensional X-33 compound; real (left), synthetic (right).

Figure 5.25: X-33 target run with 3-D models. Forty-two model objects populate the X-33 compound.

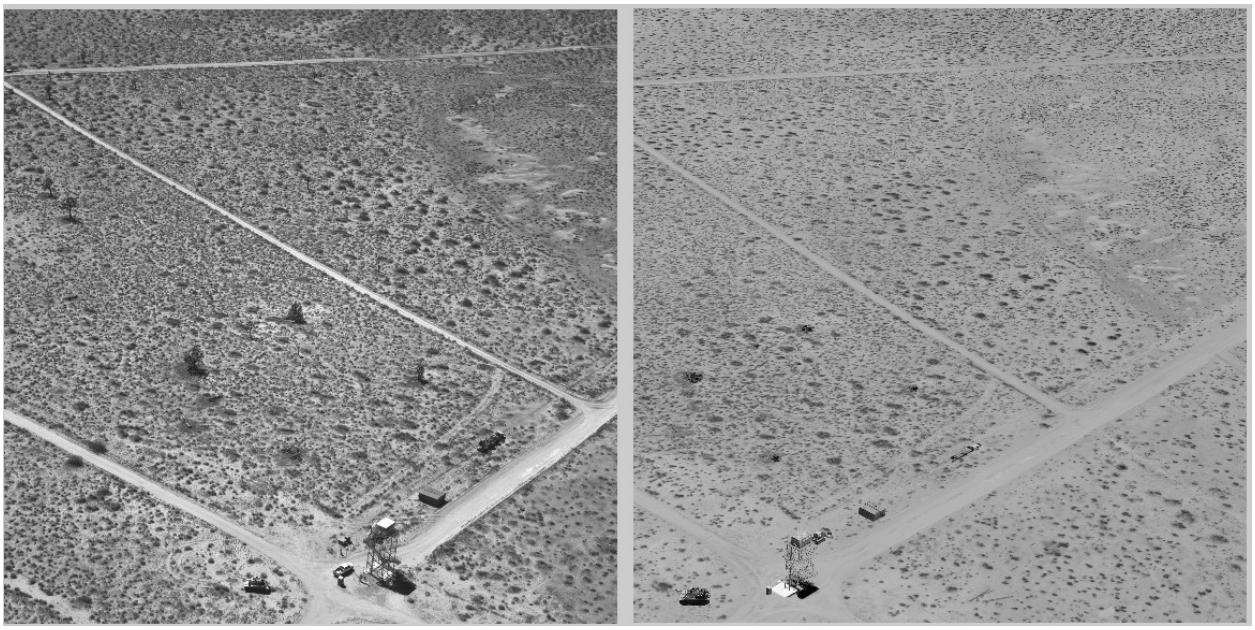


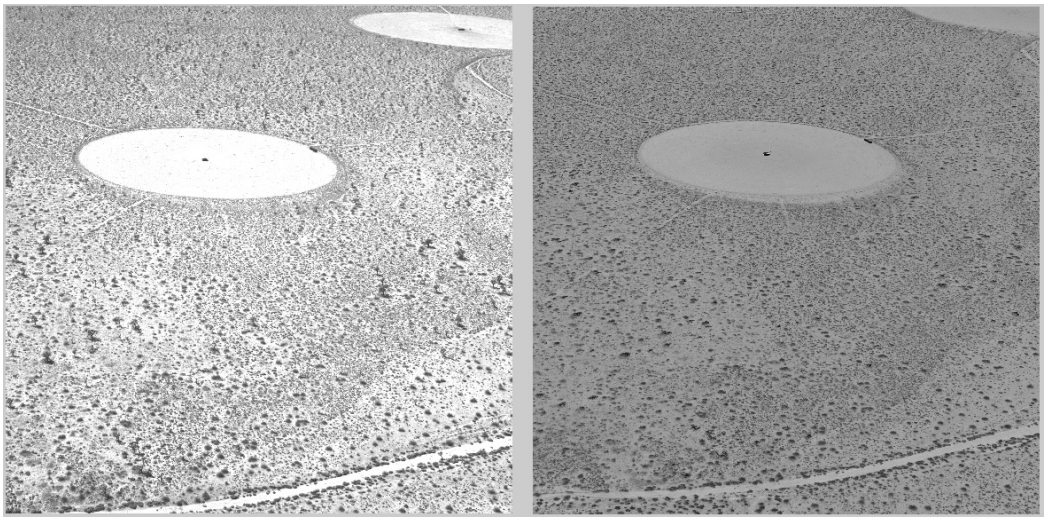
Figure 5.26: Flat trees in virtual image; real (left), synthetic (right). The visual information available in natural resources could help achieve high-level precision someday.

for verification. The 270° tank run was processed with both Sortie 1 images and data and the redo Sortie 4 images and data. Figure 5.27 provides two frames each from the real and synthetic images sets. The results, shown in Table 5.7 actually show an improved solution; the INS-only solution for the overexposed sortie also yielded a worse solution. Since the run was against the tank, which is a small dark object in a large low contrast area, more SIFT® points were found in the overexposed case.

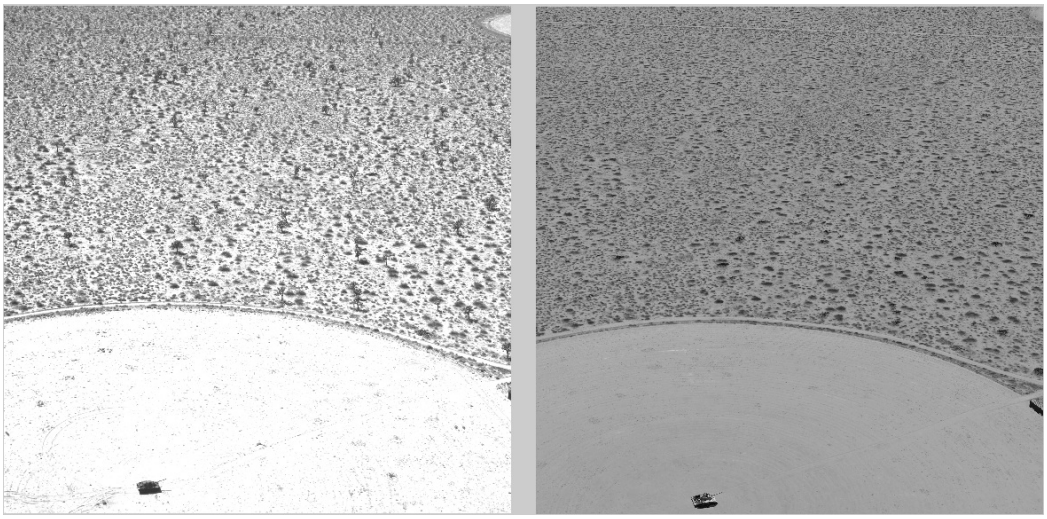
Table 5.7: Effect of overexposure and clouds in spherical error (feet). Results show improvement overexposed data for the tank target.

Case	INS error (ft)	VANSPR error (ft)
Normal settings	532	178
Overexposed	558	129

5.4.5 Effect of Bad SIFT® matching. One particular case of bad SIFT® matching was studied to include the effects of threshold changing. Figure 5.28(a) shows an end-stage run where the solution is obviously suffering. While each mea-



(a) Mid-run image of tank on PB-9.

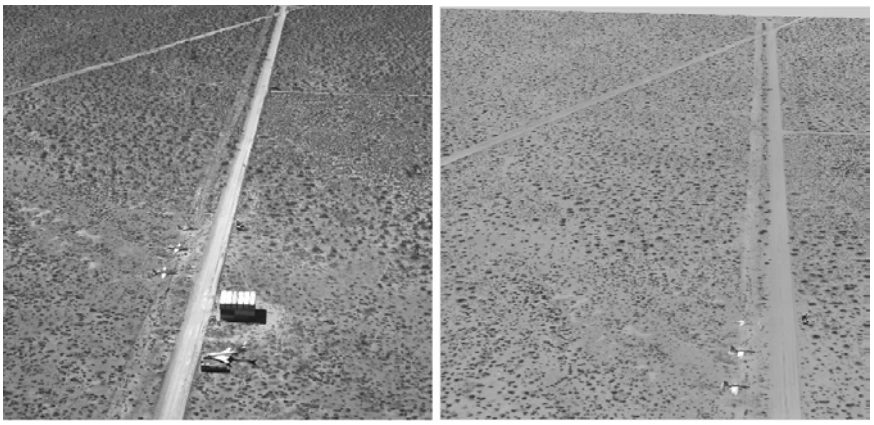


(b) End of run against tank on PB-9.

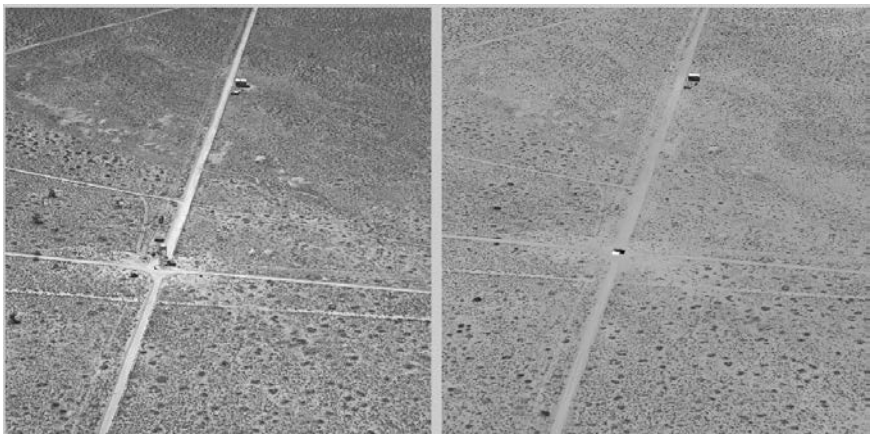
Figure 5.27: Comparison of overexposed test camera image (left) and synthetic image (right) of the same.

surement update usually consists of several hundred SIFT® points per comparison, this run had a stretch of several updates where less than 10 (which the algorithm ignores) or between 10 and 20 (which will be accepted but may be suspect) SIFT® pairs were observed. The resultant spherical error was 303 feet. Since the algorithm is intended to be autonomous, it is not desired to have to manually adjust parameters, but for investigation purposes, the SIFT® threshold was increased from the nominal 0.6 to 0.75. The result was many more SIFT® points found and an endgame positional error of 70 feet, a 77% improvement. Figure 5.28(b) and (c) show real and synthetic views with the improved SIFT® threshold. It should be noted that increasing the SIFT® threshold cannot be a *carte blanche* solution. When a relatively large number of SIFT® matches are already being found, turning up the threshold allows for less discriminating matches. Even if the matches were all perfect, adding more than required needlessly increases computational burden.

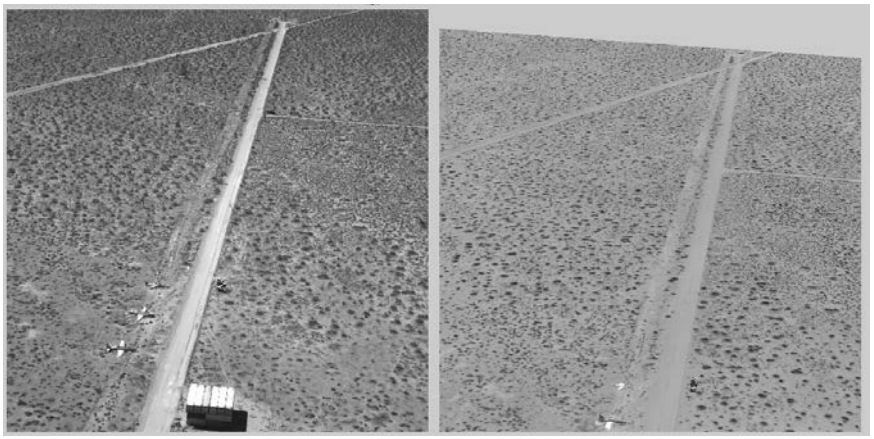
5.4.6 Summary. This was a summary of some of the some of the many effects that could be studied when cameras are incorporated into navigation solutions. Many variables can cause a substantial effect including measurement rates and scheduling, selection of the measurement noise model, run-in headings, three-dimensional effect of computer-generated models, and number of available SIFT® matches. The INS-only endgame error was 411 feet. Using $\sigma_x = \sigma_y = 20m$ for measurement uncertainty, VANSPR error was 288 feet, a 30% improvement. When the empirical $\mathbf{R}$ was used for the runs with the highest errors, the overall result became 166 feet, for a 63% improvement.



(a) Conex 225° run with low number of SIFT® matches; real (left), synthetic (right).



(b) Conex 255 with higher SIFT® matches; real (left), synthetic (right).



(c) Improved run completion due to higher SIFT® matching; real (left), synthetic (right).

Figure 5.28: Effects of increasing SIFT® threshold to aid feature point starvation. Images pairs (b) and (c) are from a second run after the SIFT® threshold was increased.

VI. Conclusions and Recommendations

This final thesis chapter will reiterate overall conclusions of the research and make recommendations for follow-on work, both near and far-term.

6.1 Conclusions

The work of this thesis presented the first application of Statistical Predictive Rendering (SPR) to the task of vision-aided navigation for a weapon's terminal guidance system. An average endgame positional error of 166 feet (using two different values of $\mathbf{R}$, was achieved compared with 411 feet for an INS-only solution over the course of a 60-second run. Proper selection of $\mathbf{R}$ proved to be a critical and elusive component of success. While this is a step towards the eventual goal of having an autonomous weapon with the flexibility of a JDAM and the accuracy of an LGB, more work is required in several areas.

The desire of autonomy was a supporting goal that drove many engineering decisions throughout the research. Many parameters could be tweaked to get better results during specific conditions, but it was desired to have a "one-size-fits-all" algorithm that could be employed in any environment. Because of this constraint, all pixel-based methods, simple corner detectors, and line and edge detection algorithms were rejected. Other compromises were made to SIFT® thresholding, update rates, etc.

Part of the intent of this research was to study the effect of navigation amongst three-dimensional objects - a realistic problem for a smart weapon dropping through an area of rugged terrain or significant urban development. Because of the inherent limitations of an airplane and the further necessarily imposed limitations of safety, this aspect of the research was not satisfactorily executed. The groundwork, however, has been laid for future work that will examine this in more detail.

While a learning curve exists for model creation through algorithm employment in its current state, some basic familiarity is all that is required for an engineer or operator to quickly start building targets and running the "plug-and-play" VANSPR

software. With a new TSPI file, inertial file, and a folder of images, the algorithm is ready to be used in minutes.

None of the assumptions stated at the beginning of this thesis were disproved. One essential assumption of experimental interest was assuming that the attitude rate values provided by the high-quality Honeywell HG1700 IMU gyros would remain accurate over the course of the 60-second data collection runs. The allowed for considerable simplification of the algorithm's estimation responsibilities.

The VANSPR algorithm was most successful against the Cowbell Tower target and least effective against the SAEC board and X-33 compound. It was also demonstrated that a weapon profile run against the X-33 compound with or without three-dimensional models produces approximately the same results.

6.2 Recommendations for Future Work

Many opportunities for improvement and expansion presented themselves throughout the research and development of the VANSPR algorithm. Some suggestions would be appropriate for near-term incorporation while others are larger in scope and would likely result from a building block approach for the purposes of academic research.

6.2.1 Near-Term Follow-up Recommendations. Suggestions for immediate follow-on work include:

- Perform analysis on non-weapon profile data that was collected. This data was not of particular interest in the context of a non-power glide weapon but is in the same format and can readily be processed by the VANSPR algorithm for further SPR navigation research. Furthermore, this analysis may provide the additional insight required for a highly-robust $\mathbf{R}matrix$.
- Make the SIFT® thresholding value adaptive and work to create a noise matrix that is more rigorously developed and adapted.

- Incorporate some means of altitude observability; refine use of the sigma point grid to to create a sigma cube or other hypergeometric shape.
- Find innovative methods to fuze both SIFT® feature point and pixel-based methods to pickup the heavy lifting when the other fails.
- Perform data collection from a vehicle that can capture images to a closing range of 1-2 meters to better simulate a weapon. A helicopter would be a perfect choice, provided that the increased vibration would not degrade image quality. The possibilities of having useable information is limited only by the resolution of the on-board camera and of the model. See Figure 6.1
- Create more accurate models with accurate terrain elevation. The constant terrain elevation assumption was a simplification for proof-of-concept for this research. However, this assumption resulted in obvious and sometimes large errors. Digital Terrain Elevation Data (DTED) could be utilized to mitigate this issue. Some caution should be given, however, to creating large models with more detail than can be handled by the computer processor, graphics processor, or MATLAB®.
- Camera boresight using a laser-based method. The flight test portion of this project intended to use a laser method for boresighting; however, an equipment failure required that a less accurate method be used that required moving a mechanical "arm" around various reference points on the interior and exterior of the aircraft. No serious degradation was noted in the end products, but it was unacceptable to have unquantifiable errors in what should have been a very precise procedure.
- Explore image processing techniques to make further use of pixel-based methods, Hough line transforms, and simple feature detection (e.g., Harris-Stephens corner detection). More advanced filtering techniques, which was beyond the scope of this thesis, may prove to be profitable to make these techniques useable.

- Incorporate additional, including non-traditional, sensors such as a flash Laser Detection and Ranging (LADAR) to enhance position information, especially in the endgame portion of the simulated weapon trajectory. The flash LADAR creates a three-dimensional image where each pixel, in addition to grayscale intensity, also possesses depth information to the object fragment it represents. This three-dimensional mapping could be treated with a variation of SPR to provide fine alignment at close range.
- Optimize the VANSPR algorithm to work real-time. Certain aspects of processing will require novel methods to achieve this goal, but immediate improvements could be expected by taking advantage of parallel processing methods with modern multi-core computers, utilizing separate on-board Graphics Processing Units (GPUs), and porting or compiling existing code into C/C^{++}.
- If feature-based methods (e.g., SIFT$^{\textcircled{R}}$) are used exclusively, undistortion equations could be applied only to feature-matched pixel coordinates instead of the entire image as done by [23].
- Incorporate a multi-hypothesis tracking filter to capture alternative branched solutions. Practically speaking, an incorrect real/virtual image match could cause a divergence that is unrecoverable in a single filter as used in the current VANSPR implementation. Such a configuration could track multiple possibilities and bring the solution back towards the true trajectory when the appropriate filter(s) begin to clearly outperform the others.
- Incorporate anticipated shadowing effects to minimize image processing confusion. This would be especially important in target environments with significant vertical development. Shadowing with VRML models is currently possible in the Google Sketchup Pro$^{\textcircled{R}}$ application as seen in Figures 4.29, 4.30, 4.31, 4.32, and 4.33. Additional programming could take time of day and observer location to calculate sun position, light angles, and resultant shadowing.

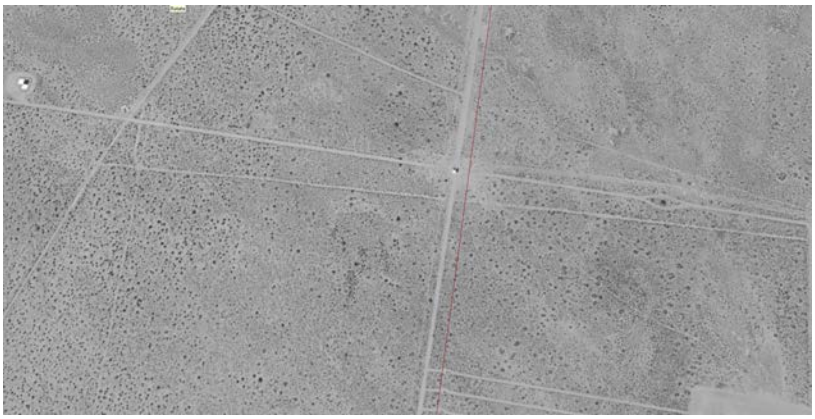
- Add two additional cameras, mounted orthogonally to the first and each other. Detecting perspective changes of distant objects by attempting to observe translational changes can be extremely challenging. Angular changes are more detectable and could be exploited by providing a means of observation in three rotation axes. As an example, viewing the stars from different locations on the same hemisphere of the Earth provides very little viewpoint change. However, even a small rotational change is observable and measurable.

6.2.2 Long-Term Follow-up Recommendations. More ambitious future research might also consider:

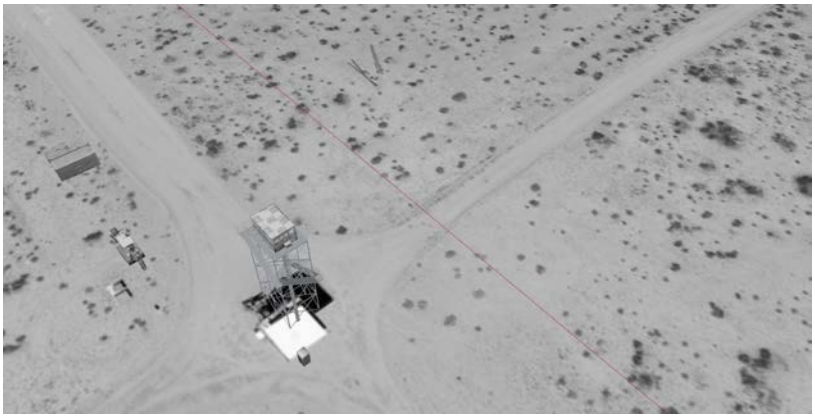
- Incorporate Artificial Intelligence (AI) algorithms that recognize classes and types of target objects during flight and make real-time decisions of priority and result maximization. Existing intelligence databases may be of use in creating VRML objects for "off-the-shelf" application.
- Develop cooperative networking of enroute weapons or vehicles that share viewpoints to refine the three-dimensional picture. Preceding weapons or vehicles can perform a "reconnaissance" role to make subsequent vehicles "smarter and smarter". Integration into future System of System (SoS) datalink networks could also make use of feeds from Global Hawks, U-2s, satellites, etc.
- Incorporate research from the field of automatic object reconstruction from images. This research was briefly mentioned in Section 2.11.4.

6.3 Summary

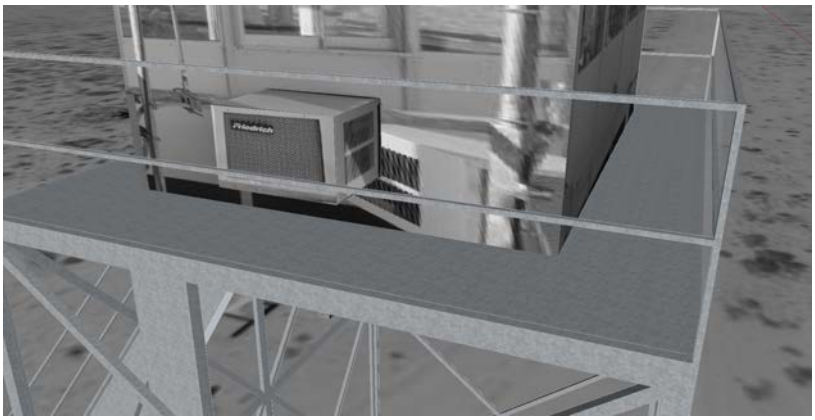
This thesis presented the development and early application of an autonomous precision weapon terminal guidance system using a tightly-coupled INS and camera. The use of statistical predictive rendering was used to provide a visual means of comparison with actual images and remove errors from the navigation solution. The goal of outperforming an INS-only solution was successfully met. However, follow-on



(a) High-altitude viewpoint of Cowbell Tower (synthetic).



(b) Close-up view of Cowbell Tower (synthetic).



(c) Reading the brand of air conditioning unit on Cowbell Tower (synthetic).

Figure 6.1: Predictive rendering zoom effects. Resolution available to the algorithm is limited only to the fidelity of the computer model.

research is required to achieve the precision necessary for integration with real-world combat systems.

Specifications

Prosilica GE	4900
Interface	IEEE 802.3 1000baseT
Resolution	4872 x 3248
Sensor	Kodak KAI-16000
Type	CCD Progressive
Sensor Size	Type 35 mm
Cell size	7.4 µm
Lens mount	F
Max frame rate at full resolution	3 fps
A/D	12 bit
On-board FIFO	32 MB
	Output
Bit depth	8/12 bit
Mono modes	Mono8, Mono16
Color modes YUV	n/a
Color modes RGB	n/a
Raw modes	Bayer8, Bayer16
	General purpose inputs/outputs (GPIOs)
TTL I/Os	1 input, 3 outputs (with galvanic isolation)
Opto-coupled I/Os	0
RS-232	1
	Power/Mass/Dimensions/Regulations
Power requirements (DC)	12 V
Power consumption (12 V)	6 W
Mass	391g (402g for GE4000C)
Body Dimensions (L x W x H in mm)	66x66x110 including connectors, w/o tripod and lens

Figure A.1: Prosilica 4900 datasheet.

Parameter	Typical Value
Architecture	Interline CCD; Progressive Scan
Total Number of Pixels	4960(H) x 3324(V) = 16.6M
Number of Effective Pixels	4904(H) x 3280(V) = 16.1M
Number of Active Pixels	4872(H) x 3248(V) = 15.8M
Pixel Size	7.4μm(H) x 7.4μm(V)
Active Image Size	36.1mm(H) x 24.0mm (H) 43.3mm(diagonal)
Aspect Ratio	3:2
Number of Outputs	1 or 2
Saturation Signal	30,000 electrons
Output Sensitivity	30μV/e
Quantum Efficiency KAI-16000-AXA (500nm)	45%
Quantum Efficiency KAI-16000-CXA R(630nm), G(540nm), B(470nm)	42%, 37%, 30%
Read Noise (f=30MHz)	16 electrons
Dark Current	$< 0.5 nA/cm^2$
Dark Current Doubling Temperature	7° C
Dynamic Range	65 dB
Charge Transfer Efficiency	0.99999
Blooming Suppression	> 100X
Smear	< -80 dB
Image Lag	< 10 electrons
Maximum Data Rate	30 MHz per channel
Package	40 pin Pin Grid Array
Cover Glass	AR coated, 2 sides

Figure A.2: Kodak KAI-16000 image sensor technical specifications.

Parameter	Value
Focal length	50 mm
Aperture range	f/1.4 – f/16(1/2 stop intervals)
Number of elements / groups	7/6
Working distance (object to sensor)	35.1 cm(1.15ft) - ∞
Angularfield* (diag. / horz. / vert.)	45 / 38 / 26 °
Max. diameter of image field	43 mm (1.7")
Flange focal length	46.5mm (1.8")
Coverage at close range	16 x 24 cm(6.3 x 9.4")
Image ratio at close range	1:6.7
Filter-thread	M 58 x 0.75
Length (without caps)**	44.8 mm(1.75")
Diameter	66 mm(2.6")
Weight	350 g (12 oz.)
Camera mount***	ZF (F bayonet)
* Referring to 35 mm format	
** from bayonet mount to filter thread when lens focused to infinity	
*** other mounts available on request	

Figure A.3: Carl Zeiss Planar T* 4/50 ZF lens technical specifications.

Bibliography

1. Baillard, C., C. Schmid, A. Zisserman, and A. Fitzgibbon. "Automatic Line Matching and 3D Reconstruction of Buildings from Multiple Views". *ISPRS Conference on Automatic Extraction of GIS Objects from Digital Imagery*. Oxford, England, 1999.

2. Baumberg, Adam. "Reliable Feature Matching Across Widely Separated Views". *IEEE Computer Society Conference on Computer Vision and Patter Recognition*. Institute of Electrical and Electronics Engineers (IEEE), Hilton Head, South Carolina, 13-15 June 2000.

3. Bay, Herbert, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. "SURF: Speeded Up Robust Features". *Computer Vision and Image Understanding (CVIU)*, Vol. 110, No. 3, pp. 346–359, 2008.

4. Beich, Jonathan W., Kevin M. Hall, Mark A. Hanson, Brett A. Pugsley, J. David Slack, and Nathan W. Taylor. *Data Collection for Autonomous Precision Weapon Terminal Guidance Using a Tightly-Coupled INS and Camera*. In-House AFFTC-TIM-10-08, Air Force Test Pilot School, Air Force Flight Test Center, Edwards AFB, CA, December 2010.

5. Beich, Jonathan W. and Michael Veth. "Tightly-Coupled Image-Aided Inertial Relative Navigation Using Statistical Predictive Rendering Techniques and a Priori World Models". *Position Location and Navigation Symposium*. Institute of Electrical and Electronics Engineers (IEEE), Indian Wells, CA, May 2010.

6. Brown, D.C. "Decentering Distortion of Lenses". *Photometric Engineering*, 444–462, Vol. 32, No.3 1966.

7. Brown, Robert Grover and Patrick Y.C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering*. John Wiley & Sons, New York, NY, third edition, 1997.

8. Canan, James W. "Air Force Technology: Change on the Horizon". *Aerospace America (AIAA) Magazine*, 28–33, November 2010.

9. Center, Air Force LeMay Doctrine. "Strategic Attack". Air Force Doctrine Document 2-1.2, June 2007.

10. Defense Mapping Agency. "Geodesy for the Layman". DMA TR80-003, December 1993.

11. Duy, Bui The and Ma Thi Chau. "A Process of Building 3D Models from Images". *VNU Journal of Science, Mathematics-Physics*, Vol. 23 9–14, 2007.

12. Ebcin, Sedat. *Tightly Integrating Optical and Inertial Sensors for Navigation Using the UKF*. Master's thesis, Graduate School of Engineering, Air

Force Institute of Technology (AETC), Wright-Patterson AFB OH, March 2008. AFIT/GE/ENG/08-09.

13. Fischler, M.A. and R.C. Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". *Communication of the ACM 24*, 381–395, 2008.

14. Geospatial Sciences Division, National Geospatial Intelligency Agency, Geodetic Surveys Branch. "PIRA Bombing and Laser Targets, Edwards AFB, California". Publication 06-1E10, September 2006.

15. Giebner, Michael G. *Tightly-Coupled Image-Aided Inertial Navigation System via a Kalman Filter*. Master's thesis, Graduate School of Engineering, Air Force Institute of Technology (AETC), Wright-Patterson AFB OH, March 2003. AFIT/GE/ENG/03-10.

16. Gonzalez, Rafael C., Richard E. Woods, and Steven L. Eddins. *Digital Image Processing Using MATLAB*. Gatesmark Publishing, Natick, MA, second edition, 2009.

17. Julier, Simon J. and Jeffrey K. Uhlmann. "Unscented Filtering and Nonlinear Estimation". *Proceedings of the IEEE*, 401–422, March, Vol. 92, No.3 2004.

18. Kayton, Myron and Walter R. Fried. *Avionics Navigation Systems*. John Wiley & Sons, Inc., New York, NY, second edition, 1997.

19. Lowe, David G. "Object Recognition from Local Scale-Invariant Features". *Proceedings of the International Conference on Computer Vision*, volume 2, 1150–1157. Corfu, Greece, September 1999.

20. Maybeck, Peter S. *Stochastic Models, Estimation, and Control, Volume 1*. Navtech Book & Software Store, Arlington, VA, reprinted edition, 1994.

21. Nakagawa, Masafumi, Ryosuke Shibasaki, and Yoshiaki Kagawa. "Fusing Stereo Linear CCD Image and Laser Range Data for Building 3D Urban Model". *Symposium on Geospatial Theory, Processing, and Applications*. University of Tokyo, Japan, 2002.

22. National Imagery and Mapping Agency. "Department of Defense World Geodetic System 1984". NIMA TR8350.2, Third Edition, January 2000.

23. Nielsen, Michael B. *Development and Flight Test of a Robust Optical-Inertial Navigation System Using Low-Cost Sensors*. Master's thesis, Graduate School of Engineering, Air Force Institute of Technology (AETC), Wright-Patterson AFB OH, March 2008. AFIT/GE/ENG/08-19.

24. Range Division, Air Force Flight Test Center. "GLite System Configuration 2B Version 2, Release 2.1 User's Manual". Edwards AFB, California, July 2006.

25. Raquet, John F. and Michael Giebner. "Navigation Using Optical Measurements of Objects at Unknown Locations". *Proceedings of ION 59th Annual Meeting/-*

CIGTF 22nd Guidance Test Symposium. Albuquerque, New Mexico, 23-25 June 2003.

26. Rousseeuw, P.J. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". *Journal of the American Statistical Association*, Vol. 79, No. 388, pp.871–880, December 1984.

27. Sellers, J. *Understanding Space: An Introduction to Astronautics*. McGraw-Hill Primis Custom Publishing, second edition, 2003.

28. Shi, J. and C. Tomasi. "Good Features to Track". *9th IEEE Conference on Computer Vision and Pattern Recognition*. Institute of Electrical and Electronics Engineers (IEEE), June 1994.

29. Strecha, Christoph, Tinne Tuytelaars, and Luc Van Gool. "Dense Matching of Multiple Wide-Baseline Views". *Computer Vision*, Vol 2, 1194–1201, October 2003.

30. The MathWorks Inc. "Simulink 3D Animation 5 User's Manual". electronic format, September 2010.

31. Titterton, D. and J. Weston. *Strapdown Inertial Navigation Technology*. The American Institute of Aeronautics and Astronautics, second edition, 2004.

32. Tuytelaars, Tinne and Luc Van Gool. "Wide Baseline Stereo Matching based on Local, Affinely Invariant Regions". *British Machine Vision Conference*. University of Bristol, UK, 11-14 September 2000.

33. United States Air Force. "Joint Direct Attach Munitions, GBU 31/32/38". Fact-sheet, January 2006.

34. Veth, M. and J. Raquet. "Alignment and Calibration of Optical and Inertial Sensors Using Stellar Observations". *Proceedings of ION GNSS 2005, Long Beach, CA*, 2494–2503, September 2005.

35. Veth, Michael. *Fusion of Imaging and Inertial Sensors for Navigation*. Ph.D. dissertation, Graduate School of Engineering, Air Force Institute of Technology (AETC), Wright-Patterson AFB OH, September 2006. AFIT/DS/ENG/06-09.

36. Veth, Michael and John F. Raquet. *Two-Dimensional Stochastic Projections for Tight Integration of Optical and Inertial Sensors for Navigation*. paper, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, 2007.

37. Weaver, Adam D. *Using Predictive Rendering as a Vision-Aided Technique for Autonomous Aerial Refueling*. Master's thesis, Graduate School of Engineering, Air Force Institute of Technology (AETC), Wright-Patterson AFB OH, March 2009. AFIT/GE/ENG/09-45.

Vita

Major Jonathan W. Beich graduated with an Associate of Science in Microcomputer Science from North Dakota State College of Science in 1995, followed by a Bachelor of Science in Electrical Engineering with a minor in physics from North Dakota State University (NDSU) in 1999. While at NDSU, he served as a cadet wing commander of Air Force Reserve Officer Training Corps (AFROTC) Detachment 610, and received a commission as a Second Lieutenant in the United States Air Force upon graduation.

Major Beich's first assignment was Joint Specialized Undergraduate Navigator Training (JSUNT) at NAS Pensacola, Florida, where he received his navigator wings in February 2001. After completing B-52H initial qualification at Barksdale AFB, Louisiana, he was assigned to the 23d Bomb Squadron, Minot AFB, North Dakota. While there, Major Beich served as an instructor radar navigator, scheduler, and assistant flight commander and flew 35 combat missions in support of Operations Enduring Freedom and Iraqi Freedom over the course of four deployments.

In 2006, Major Beich was reassigned as the B-52H Deputy Test Director, 31st Test and Evaluation Squadron, Edwards AFB, California to perform operational flight test and developmental flight test support. In September 2007, he completed a Master of Science degree in Space Studies from American Military University while deployed to Camp As Sayliyah, Qatar where he was serving as Officer in Charge of an Arabic media translation cell. In September 2008, he began the Joint AFIT/TPS program at Wright-Patterson AFB, Ohio. After completing all coursework at the Air Force Institute of Technology (AFIT) in December 2009, he was reassigned to the Air Force Test Pilot School (TPS) at Edwards AFB, California, which he graduated from in December 2010 as a member of Class 10A. This thesis is the final requirement for his Master of Science degree in Electrical Engineering from AFIT.

Major Beich will next be assigned to the 419th Flight Test Squadron, Edwards AFB, California, to serve as an Experimental Test Navigator in the B-52H and B-1B.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
24–03–2011	Master's Thesis	Sept 2008 — Mar 2011

4. TITLE AND SUBTITLE

Vision-Aided Autonomous Precision Weapon Terminal Guidance Using a Tightly-Coupled INS and Predictive Rendering Techniques

5a. CONTRACT NUMBER

DACA99–99–C–9999

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Jonathan W. Beich, Maj, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GE/ENG/11-42

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Draper Laboratories, Perception Systems Group (Paul DeBitetto)
555 Technology Square
Cambridge, MA 02139
((617) 258-2468; pdebitetto@draper.com)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This thesis documents the development of the Vision-Aided Navigation using Statistical Predictive Rendering (VANSPR) algorithm which seeks to enhance the endgame navigation solution possible by inertial measurements alone. The eventual goal is a precision weapon that does not rely on GPS, functions autonomously, thrives in complex 3-D environments, and is impervious to jamming. The predictive rendering is performed by viewpoint manipulation of computer-generated of target objects. A navigation solution is determined by an Unscented Kalman Filter (UKF) which corrects positional errors by comparing camera images with a collection of statistically significant virtual images. Results indicate that the test algorithm is a viable method of aiding an inertial-only navigation system to achieve the precision necessary for most tactical strikes. On 14 flight test runs, the average positional error was 166 feet at endgame, compared with an inertial-only error of 411 feet.

15. SUBJECT TERMS

statistical predictive rendering, unscented Kalman filter, image-aided navigation, tightly-coupled INS and camera system, feature-detection, precision weapon

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Michael J. Veth, Lt Col, USAF (ENG)
U	U	U	UU	199	19b. TELEPHONE NUMBER *(include area code)* (850) 882-4667; michael.veth@eglin.af.mil